BAB II

TINJAUAN PUSTAKA

II.1 Text Editor

Text editor adalah program yang digunakan untuk mengetik teks dalam komputer, teks yang diketik dengan menggunakan *text editor* tidak ditambahi pengaturan format apapun, tidak diatur jenis hurufnya, tidak diatur spasinya, tidak diatur rata kanan-kirinya, dsb. Semua teks disimpan sebagai deretan karakter ASCII (*American Standart Code for Information Interchange*) saja. (Junaidi; 2005 : 34)

Text Editor merupakan sebuah software aplikasi atau program yang memungkinkan penggunanya untuk membuat, mengubah, atau mengedit file text (plain text). Text Editor dapat digunakan untuk membuat program komputer, mengedit source code bahasa pemograman serta mengubah halaman web atau template web design. Aplikasi ini secara umum digunakan untuk tujuan pemograman, bukan untuk pembuatan dokumen seperti fungsinya dimasa lalu. Hanya text biasa alias plain text saja yang dapat dimasukkan pada sebuah aplikasi text editor, berkebalikan dengan aplikasi word processing (pengolah kata) atau aplikasi *rich text editor* yang dapat digunakan untuk mengelola *formatted text.* Program tersebut dapat menghadirkan fungsi khusus seperti *bold, italic,* dan pengubahan ukuran font dan juga jenis font. Aplikasi text editor disisi lain hanya menampilkan semua karakter yang terlihat didalam file, atau dengan kata lain tidak memiliki kemampuan untuk memformat text dan juga tidak dapat menambahkan gambar atau konten lainnya.

II.2 Web Browser

Web Browser atau internet browser adalah sebuah aplikasi perangkat lunak untuk melintasi, mengambil, dan menyajikan sumber informasi di World Wide Web (www). Sumber informasi diidentifikasi dengan Uniform Resource Identifier (URL) termasuk sebuah halaman Web, gambar, video, atau bagian lain dari konten web. (Rahmad Rafiudin; 2002 : 173)

Jenis browser saat ini semakin banyak dan berkembang dengan pesat diantaranya adalah *Internet Explorer, Mozilla Firefox, Opera, Google Chrome, Safari, Netscape, Flock, Avant Browse*r, dan lain lain.

World Wide Web atau *Web* adalah sebuah layanan yang didapat oleh pemakai komputer apabila komputernya tersambung dengan internet. Dengan web pengguna internet lainnya dapat berkomunikasi, bertukar data, dan bertinteraksi tanpa harus berpindah atau beranjak dari tempat mana internet tersebut dapat diakses. (Andi Setiawan; 2004 : 15).

II.3 Kriptografi

Kriptografi pada awalnya dijabarkan sebagai ilmu yang mempelajari bagaimana menyembunyikan pesan. Namun pada pengertian modern kriptografi adalah ilmu yang bersandarkan pada teknik matematika yang berurusan dengan keamanan informasi seperti kerahasiaan, keutuhan data dan otentikasi entitas.
(Rifki Sadikin ;2012: 9).

II.3.1 Teori Dasar Kriptografi

Kriptografi (cryptography) berasal dari bahasa Yunani “cryptos” artinya“secret” (rahasia), sedangkan “graphein” artinya “writing” (tulisan). Jadi, kriptografi berarti “secret writing” (tulisan rahasia). Ada beberapa definisi kriptografi yang telah dikemukakan di dalam beberapa literatur. Definisi yang dipakai di dalam buku-buku yang lama (sebelum tahun 1980-an) menyatakan bahwa kriptografi adalah ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. Definisi ini mungkin cocok pada masa lalu di mana kriptografi digunakan untuk keamanan komunikasi penting seperti komunikasi di kalangan militer, diplomat dan mata-mata. Namun saat ini kriptografi lebih dari sekedar privacy, tetapi juga tujuan data integrity, authentication dan non repudiation. (Rinaldi Munir ; 2004)

Ada 4 (empat) tujuan mendasar dari ilmu kriptografi ini yang juga merupakan aspek keamanan informasi yaitu:

1. Kerahasiaan (confidentiality), adalah layanan yang ditujukan untuk menjaga agar pesan tidak dapat dibaca oleh pihak-pihak yang tidak berhak. Di dalam kriptografi, layanan ini direalisasikan dengan menyandikan pesan menjadi cipherteks.
2. Integritas data (data integrity), adalah layanan yang menjamin bahwa pesan masih asli atau belum pernah dimanipulasi selama pengiriman.

Dengan kata lain, aspek keamanan ini dapat diungkapkan sebagai pertanyaan: “Apakah pesan yang diterima masih asli atau tidak mengalami perubahan?”. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi manipulasi pesan oleh pihak-pihak yang tidak

berhak, antara lain penyisipan, penghapusan, dan pensubstitusian data lain ke dalam pesan yang sebenarnya.

3. Otentikasi (authentication), adalah layanan yang berhubungan dengan identifikasi, baik mengindentifikasi kebenaran pihak-pihak yang berkomunikasi (user authentication atau entity authentication) maupun mengindentifikasi kebenaran sumber pesan (data origin authencation). Pihak yang saling berkomunikasi harus dapatmengotentikasi satu sama lain sehingga dapat memastikan sumber pesan. Pesan yang dikirim melalui saluran komunikasi juga harus diotentikasi asalnya. Dengan kata lain, aspek keamanan ini dapat diungkapkan sebagai pertanyaan: “Apakah pesan yang diterima benar-benar berasal dari pengirim yang benar?”.
4. Nirpenyangkal (non-repudiation), adalah layanan untuk mencegah entitas yang berkomunikasi melakukan penyangkalan, yaitu pengirim pesan menyangkal melakukan pengiriman atau penerima pesan menyangkal telah menerima pesan.

II.3.2 Algoritma Kriptografi

Algoritma kriptografi disebut juga cipher yaitu aturan untuk enchipering dan dechipering, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Beberapa cipher memerlukan algoritma yang berbeda untuk enciphering dan dechiphering. Keamanan algoritma kriptografi sering diukur dari banyaknya kerja yang dibutuhkan untuk memecahkan cipherteks menjadi plainteks tanpa mengetahui kunci yang digunakan. Apabila semakin banyak

proses yang diperlukan berarti juga semakin lama waktu yang dibutuhkan, maka semakin kuat algoritma tersebut dan semakin aman digunakan untuk menyandikan pesan. Algoritma kriptografi terdiri dari fungsi dasar yaitu:

1. *Enkripsi*, merupakan hal yang sangat penting dalam kriptografi yang merupakan pengamanan data yang dikirimkan terjaga rahasianya, pesan asli disebut plainteks yang dirubah menjadi kode-kode yang tidak dimengerti. Enkripsi bisa diartikan dengan cipher atau kode.
2. *Dekripsi*, merupakan kebalikan dari enkripsi, pesan yang telah dienkripsi dikembalikan kebentuk asalnya (plainteks) disebut dengan dekripsi pesan. Algoritma yang digunakan untuk dekripsi tentu berbeda dengan yang digunakan untuk enkripsi.
3. *Kunci*, yang dimaksud di sini adalah kunci yang dipakai untuk melakukan enkripsi dan dekripsi, kunci terbagi jadi 2 (dua) bagian yaitu kunci pribadi (private key) dan kunci umum (public key).

Konsep matematis yang mendasari kriptografi adalah relasi antara 2 (dua) buah himpunan yaitu himpunan yang berisi elemen-elemen plainteks dan himpunan yang berisi cipherteks. Enkripsi dan dekripsi merupakan fungsi yang memetakan elemen-elemen antar kedua himpunan tersebut. Misalkan P menyatakan plainteks dan C menyatakan cipherteks, maka fungsi enkripsi E memetakan P ke C,

$$E(P)=C$$

Dan fungsi dekripsi D memetakan C ke P,

$$D(C)=P$$

Karena proses enkripsi kemudian dekripsi mengembalikan pesan ke pesan asal, maka kesama berikut harus benar,

$$D(E(P))=P$$

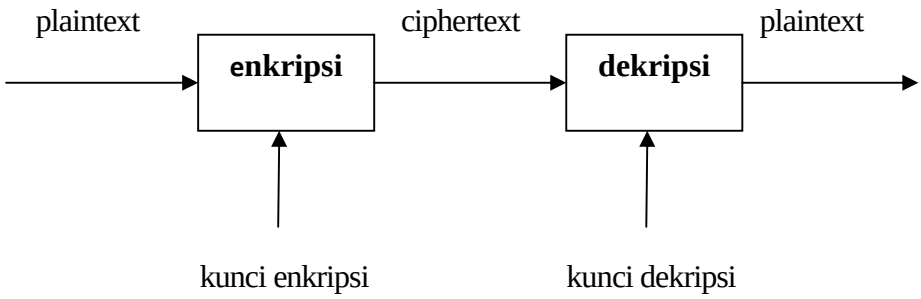
Kriptografi modern menggunakan kunci, yang dalam hal ini algoritma tidak lagi dirahasiakan, tetapi kunci harus dijaga kerahasiaannya. Kunci adalah parameter yang digunakan untuk transformasi enciphering dan dechipering. Kunci biasanya berupa string atau deretan bilangan. Jika menggunakan kunci K, maka fungsi enkripsi dan dekripsi dapat ditulis sebagai

$$E_K(P)=C \text{ dan } D_K(C)=P$$

dan ke dua fungsi ini memenuhi

$$D_K(E_K(P))=P$$

Gambar II.1 dibawah memperlihatkan skema enkripsi dan dekripsi dengan menggunakan kunci.



Gambar II.1 Skema Enkripsi dan Dekripsi

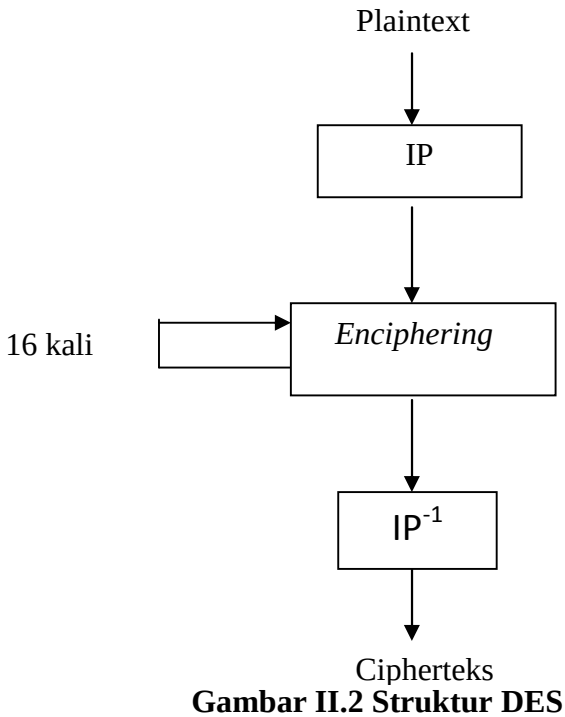
II.4. Algoritma Kriptografi Data Encryption Standard (DES)

Data Encryption Standard (DES) merupakan algoritma enkripsi yang paling banyak dipakai di dunia. DES diadopsi oleh NIST (National Institue of Standard and Technology) sebagai standar pengolahan informasi Federal AS. Secara umum DES terbagi menjadi tiga kelompok, yaitu pemrosesan kunci,enkripsi data 64 bit, dan dekripsi data 64 bit yang mana satu kelompok saling berinteraksi satu dengan yang lainnya.

DES termasuk ke dalam sistem kriptografi simetri dan tergolong jenis *cipher* blok. DES beroperasi pada ukuran blok 64 bit. DES mengenkripsikan 64 bit plainteks menjadi 64 bit cipherteks dengan menggunakan 56 bit kunci internal (*internal key*) atau upa-kunci (*subkey*). Kunci internal dibangkitkan dari kunci eksternal (*external key*) yang panjangnya 64 bit. (Rifki sadikin; 2012: 123)

II.4.1 Struktur DES

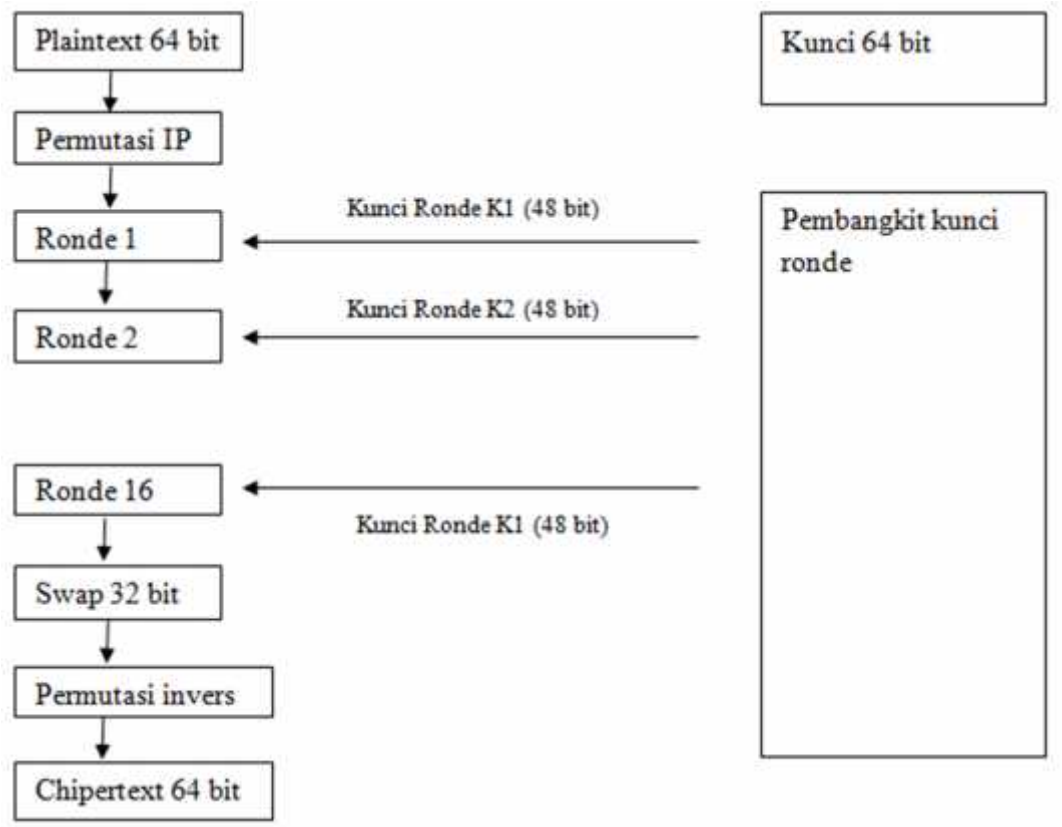
DES merupakan salah satu contoh sandi fiestel, sehingga struktur sandi DES memiliki struktur yang sama dengan sandi fiestel. Dengan pengkhususan: panjang blok DES adalah 64 bit jadi ukuran teks asli dan teks sandi adalah 64 bit. Ukuran kunci DES adalah 64 bit dan ukuran kunci ronde adalah 48 bit dan jumlah ronde pada DES adalah 16 ronde. Struktur enkripsi sandi DES diberikan oleh Gambar II.9:



Gambar II.2 Struktur DES

1. Blok plainteks dipermutasi dengan matriks permutasi awal (*initial permutation* atau IP).
2. Hasil permutasi awal kemudian di-*enciphering*- sebanyak 16 kali (16 putaran). Setiap putaran menggunakan kunci internal yang berbeda.
3. Hasil enciphering kemudian dipermutasi dengan matriks permutasi balikan (*invers initial permutation* atau IP^{-1}) menjadi blok cipherteks.

Di dalam proses *enciphering*, blok plainteks terbagi menjadi dua bagian, kiri (*L*) dan kanan (*R*), yang masing-masing panjangnya 32 bit. Kedua bagian ini masuk ke dalam 16 putaran DES.



Gambar II.3 Struktur Sandi DES

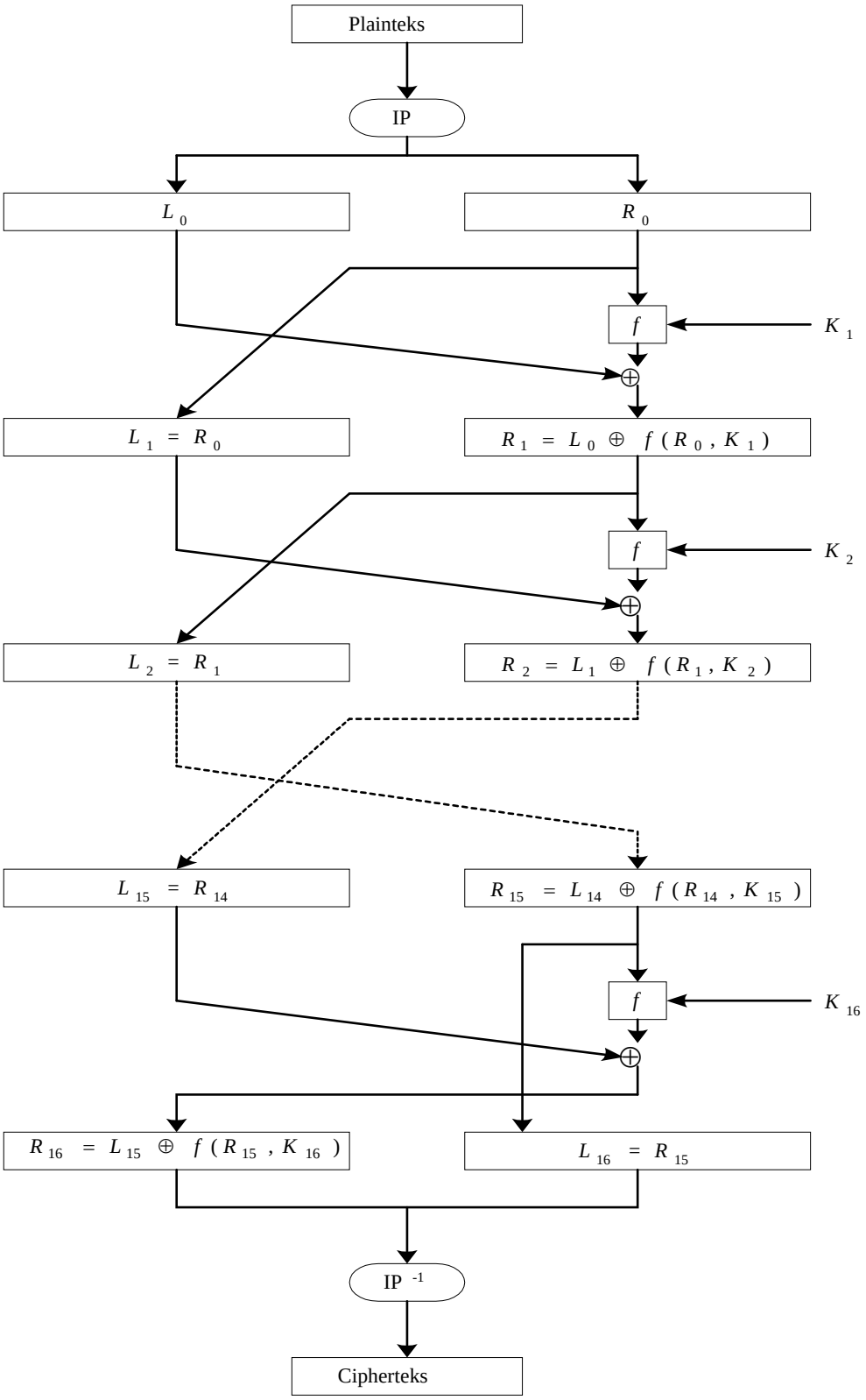
Pada setiap putaran *i*, blok *R* merupakan masukan untuk fungsi transformasi yang disebut *f*. Pada fungsi *f*, blok *R* dikombinasikan dengan kunci internal K_i. Keluaran dai fungsi *f* di-XOR-kan dengan blok *L* untuk mendapatkan blok *R* yang baru. Sedangkan blok *L* yang baru langsung diambil dari blok *R* sebelumnya. Ini adalah satu putaran DES.

Secara matematis, satu putaran DES dinyatakan sebagai

$L_i = R_{i-1}$

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

Gambar II.11 memperlihatkan skema algoritma DES yang lebih rinci.



Gambar II.4 Skema Algoritma DES lebih rinci

II.4.2 Permutasi Awal

Sebelum putaran pertama, terhadap blok plainteks dilakukan permutasi awal (*initial permutation* atau IP). Tujuan permutasi awal adalah mengacak plainteks sehingga urutan bit-biit di dalamnya berubah. Pengacakan dilakukan dengan menggunakan matriks permutasi awal berikut ini:

Tabel II.1. Matriks Permutasi Awal

58	50	42	34	26	18	10	2	60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6	64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1	59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5	63	55	47	39	31	23	15	7

Cara membaca tabel/matriks di atas: dua *entry* ujung kiri atas (58 dan 50) berarti:

"pindahkan bit ke-58 ke posisi bit 1"

"pindahkan bit ke-50 ke posisi bit 2", dst

II.4.3 Pembangkitan Kunci Internal

Karena ada 16 putaran, maka dibutuhkan kunci internal sebanyak 16 buah, yaitu K_1, K_2, …, K_{16}. Kunci-kunci internal ini dapat dibangkitkan sebelum proses enkripsi atau bersamaan dengan proses enkripsi. Kunci internal dibangkitkan dari kunci eksternal yang diberikan oleh pengguna. Kunci eksternal panjangnya 64 bit atau 8 karakter. Misalkan kunci eksternal yang tersusun dari 64 bit adalah *K*.

Kunci eksternal ini menjadi masukan untuk permutasi dengan menggunakan matriks permutasi kompresi PC-1 sebagai berikut:

Tabel II.2 Matriks Permutasi Kompresi PC-1

57	49	41	33	25	17	9	1	58	50	42	34	26	18
10	2	59	51	43	35	27	19	11	3	60	52	44	36
63	55	47	39	31	23	15	7	62	54	46	38	30	22
14	6	61	53	45	37	29	21	13	5	28	20	12	4

Dalam permutasi ini, tiap bit kedelapan (*parity bit*) dari delapan *byte* kunci diabaikan. Hasil permutasinya adalah sepanjang 56 bit, sehingga dapat dikatakan panjang kunci DES adalah 56 bit.Selanjutnya, 56 bit ini dibagi menjadi 2 bagian, kiri dan kanan, yang masing-masing panjangnya 28 bit, yang masing-masing disimpan di dalam C_0 dan D_0:

C_0: berisi bit-bit dari *K* pada posisi

57, 49, 41, 33, 25, 17, 9, 1, 58, 50, 42, 34, 26, 18

10, 2, 59, 51, 43, 35, 27, 19, 11, 3, 60, 52, 44, 36

D_0: berisi bit-bit dari *K* pada posisi

63, 55, 47, 39, 31, 23, 15, 7, 62, 54, 46, 38, 30, 22

14, 6, 61, 53, 45, 37, 29, 21, 13, 5, 28, 20, 12, 4

Selanjutnya, kedua bagian digeser ke kiri (*left shift*) sepanjang satu atau dua bit bergantung pada tiap putaran. Operasi pergeseran bersifat *wrapping* atau *round-shift*. Jumlah pergeseran pada setiap putaran ditunjukkan pada Tabel 1 sbb:

Tabel II.3. Jumlah pergeseran bit pada setiap putaran

Putaran, *i*	Jumlah pergeseran bit
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Misalkan (C_i, D_i) menyatakan penggabungan C_i dan D_i. (C_{i+1}, D_{i+1}) diperoleh dengan menggeser C_i dan D_i satu atau dua bit. Setelah pergeseran bit, (C_i, D_i) mengalami permutasi kompresi dengan menggunakan matriks PC-2 berikut:

Tabel II.4. Matrik Permutasi PC-2

14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

Dengan permutasi ini, kunci internal K_i diturunkan dari (C_i, D_i) yang dalam hal ini K_i merupakan penggabungan bit-bit C_i pada posisi:

14, 17, 11, 24, 1, 5, 3, 28, 15, 6, 21, 10

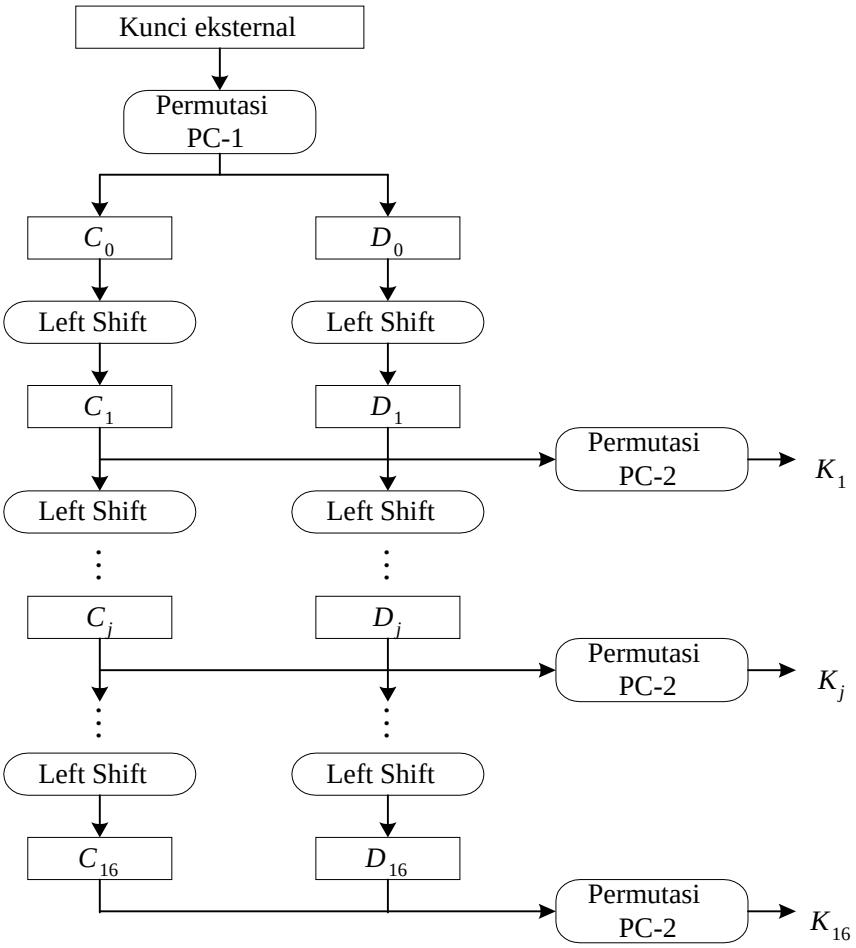
23, 19, 12, 4, 26, 8, 16, 7, 27, 20, 13, 2

dengan bit-bit D_i pada posisi:

41, 52, 31, 37, 47, 55, 30, 40, 51, 45, 33, 48

44, 49, 39, 56, 34, 53, 46, 42, 50, 36, 29, 32

Jadi, setiap kunci internal K_i mempunyai panjang 48 bit.. Bila jumlah pergeseran bit-bit pada Tabel 1 dijumlahkan semuanya, maka jumlah seluruhnya sama dengan 28, yang sama dengan jumlah bit pada C_i dan D_i. Karena itu, setelah putaran ke-16 akan didapatkan kembali $C_{16} = C_0$ dan $D_{16} = D_0$. Proses pembangkitan kunci-kunci internal ditunjukkan pada Gambar II.12 berikut:



Gambar II.5. Proses pembangkitan kunci-kunci internal DES

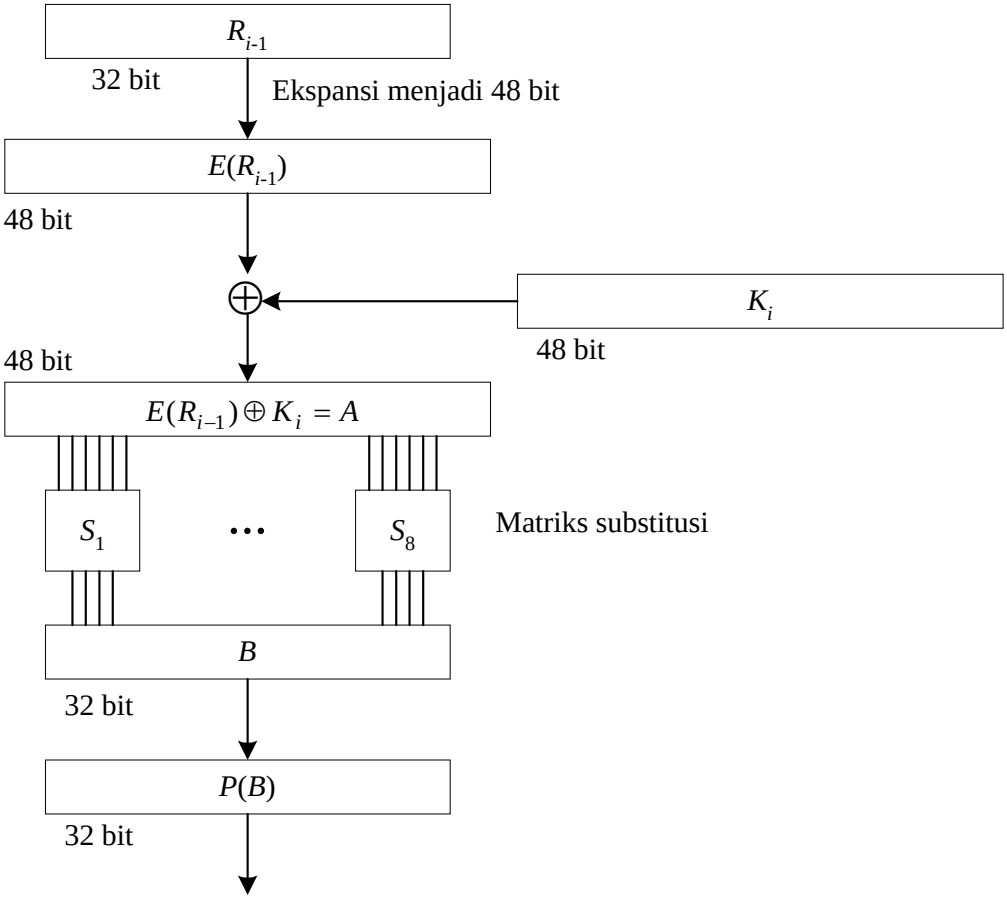
II.4.4 Enciphering

Proses *enciphering* terhadap blok plainteks dilakukan setelah permutasi awal (lihat Gambar 1). Setiap blok plainteks mengalami 16 kali putaran *enciphering* (lihat Gambar 2). Setiap putaran *enciphering* merupakan jaringan Feistel yang secara matematis dinyatakan sebagai

$$L_i = R_{i-1}$$

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

Diagram komputasi fungsi *f* diperlihatkan pada Gambar II.13. berikut:



Gambar II.6. Rincian komputasi fungsi *f*

E adalah fungsi ekspansi yang memperluas blok R_{i-1} yang panjangnya 32-bit menjadi blok 48 bit. Fungsi ekspansi direalisasikan dengan matriks permutasi ekspansi sbb:

Tabel II.5. Matriks Permutasi Ekspansi

32	1	2	3	4	5	4	5	6	7	8	9
8	9	10	11	12	13	12	13	14	15	16	17
16	17	18	19	20	21	20	21	22	23	24	25
24	25	26	27	28	29	28	29	30	31	32	1

Selanjutnya, hasil ekpansi, yaitu $E(R_{i-1})$, yang panjangnya 48 bit di-XOR-kan dengan K_i yang panjangnya 48 bit menghasilkan vektor *A* yang panjangnya 48-bit:

$$E(R_{i-1}) \oplus K_i = A$$

Vektor *A* dikelompokkan menjadi 8 kelompok, masing-masing 6 bit, dan menjadi masukan bagi proses substitusi. Proses substitusi dilakukan dengan menggunakan delapan buah kotak-S (*S-box*), S_1 sampai S_8. Setiap kotak-S menerima masukan 6 bit dan menghasilkan keluaran 4 bit. Kelompok 6-bit pertama menggunakan S_1, kelompok 6-bit kedua menggunakan S_2, dan seterusnya.

(*cara pensubstitusian dengan kotak-S sudah dijelaskan pada materi "Prinsip-prinsip Perancangan* Cipher *Blok"*)

Kedelapan kotak-S tersebut adalah:

Tabel II.6. Kotak -S

S_1:

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2:

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3:

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4:

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5:

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	16
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6:

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7:

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8:

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Keluaran proses substitusi adalah vektor *B* yang panjangnya 48 bit. Vektor *B* menjadi masukan untuk proses permutasi. Tujuan permutasi adalah untuk mengacak hasil proses substitusi kotak-S. Permutasi dilakukan dengan menggunakan matriks permutasi *P* (*P-box*) sbb:

Tabel II.7. Permutasi P(P-Box)

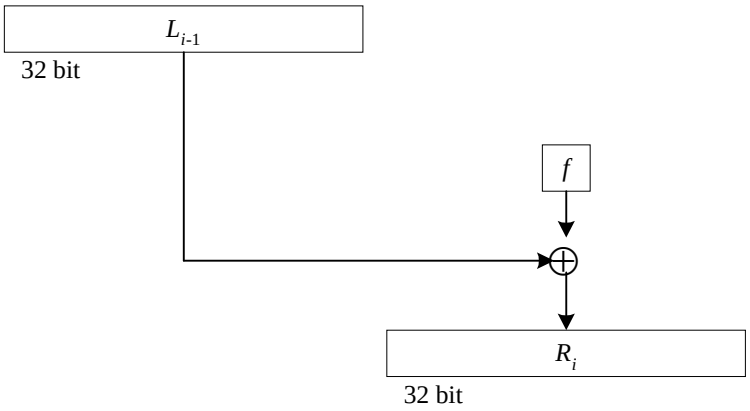
16	7	20	21	29	12	28	17	1	15	23	26	5	8	31	10
2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

Bit-bit *P*(*B*) merupakan keluaran dari fungsi *f*. Akhirnya, bit-bit *P*(*B*) di-XOR-kan dengan L_{i-1} untuk mendapatkan R_i (lihat Gambar II.7):

$$R_i = L_{i-1} \oplus P(B)$$

Jadi, keluaran dari putaran ke-*i* adalah

$$(L_i, R_i) = (R_{i-1}, L_{i-1} \oplus P(B))$$



Gambar II.7. Skema perolehan R_i

II.4.5 Permutasi Terakhir (*Inverse Initial Permutation*)

Permutasi terakhir dilakukan setelah 16 kali putaran terhadap gabungan blok kiri dan blok kanan. Proses permutasi menggunakan matriks permutasi awal balikan (*inverse initial permutation* atau IP^{-1}) sbb:

Tabel II.8 *inverse initial permutation* atau IP^{-1}

40	8	48	16	56	24	64	32	39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30	37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28	35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26	33	1	41	9	49	17	57	25

II.4.6 Dekripsi

Proses dekripsi terhadap cipherteks merupakan kebalikan dari proses enkripsi. DES menggunakan algoritma yang sama untuk proses enkripsi dan dekripsi. Jika pada proses enkripsi urutan kunci internal yang digunakan adalah K_1, K_2, …, K_{16}, maka pada proses dekripsi urutan kunci yang digunakan adalah K_{16}, K_{15}, …, K_1. Untuk tiap putaran 16, 15, …, 1, keluaran pada setiap putaran *deciphering* adalah

$$L_i = R_{i-1}$$

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

yang dalam hal ini, (R_{16}, L_{16}) adalah blok masukan awal untuk deciphering. Blok (R_{16}, L_{16}) diperoleh dengan mempermutasikan cipherteks dengan matriks permutasi IP^{-1}. Pra-keluaran dari *deciphering* adalah adalah (L_0, R_0). Dengan permutasi awal IP akan didapatkan kembali blok plainteks semula.

Di Tinjau kembali proses pembangkitan kunci internal pada Gambar 4. Selama *deciphering*, K_{16} dihasilkan dari (C_{16}, D_{16}) dengan permutasi PC-2. Tentu saja (C_{16}, D_{16}) tidak dapat diperoleh langsung pada permulaan *deciphering*. Tetapi karena (C_{16}, D_{16}) = (C_0, D_0), maka K_{16} dapat dihasilkan dari (C_0, D_0) tanpa perlu lagi melakukan pergeseran bit. Catatlah bahwa (C_0, D_0) yang merupakan bit-bit dari kunci eksternal *K* yang diberikan pengguna pada waktu dekripsi.

Selanjutnya, K_{15} dihasilkan dari (C_{15}, D_{15}) yang mana (C_{15}, D_{15}) diperoleh dengan menggeser C_{16} (yang sama dengan C_0) dan D_{16} (yang sama dengan C_0) satu bit ke kanan. Sisanya, K_{14} sampai K_1 dihasilkan dari (C_{14}, D_{14}) sampai (C_1, D_1). Catatlah bahwa (C_{i-1}, D_{i-1}) diperoleh dengan menggeser C_i dan D_i dengan cara yang sama seperti pada Tabel 1, tetapi pergeseran kiri (*left shift*) diganti menjadi pergeseran kanan (*right shift*).

II.4.7 Mode DES

DES dapat dioperasikan dengan mode ECB, CBC, OFB, dan CFB. Namun karena kesederhanaannya, mode ECB lebih sering digunakan pada paket program komersil meskipun sangat rentan terhadap serangan. Mode CBC lebih kompleks daripada EBC namun memberikan tingkat keamanan yang lebih bagus daripada mode EBC. Mode CBC hanya kadang-kadang saja digunakan.

II.4.8 Implementasi *Hardware* dan *Software* DES

DES sudah diimplementasikan dalam bentuk perangkat keras. Dalam bentuk perangkat keras, DES diimplementasikan di dalam *chip*. Setiap detik *chip* ini dapat mengenkripsikan 16,8 juta blok (atau 1 gigabit per detik). Implementasi DES ke dalam perangkat lunak dapat melakukan enkripsi 32.000 blok per detik (pada komputer *mainframe* IBM 3090).

II.5 RC2 atau RSA (Rivest, Shamir, Adleman)

Penemu pertama algoritma kriptografi kunci asimetri adalah Clifford Cocks, James H. Ellis dan Malcolm Williamson (sekelompok ahli matematika yang bekerja untuk *United Kindom's Goverment Communication Head Quarters*. ages rahasia Inggris) pada awal tahun 1970. Pada waktu itu temuan itu dipublikasikan dan fakta mengenai temuan tersebut tetap menjadi rahasia hingga tahun 1997.

Algoritma kriptografi kunci asimetri untuk pertama kalinya dipublikasikan pada tahun 1976 oleh Whitfield Diffie dan Martin Hellman. Dua orang tersebut merupakan ilmuwan dari Stanford University, yang membahas metode pendistribusian kunci rahasia melalui saluran komunikasi umum (publik), yang kemudian metode tersebut dikenal dengan metode pertukaran kunci Diffie-Hellman *(Diffle-Hellinan Kee Exchange).*

Ide awal Clifford Cocks diternukan kembali oleh sekelompok ilmuwan dari Massachussets Institute of Technology pads tahun 1977. sekelompok orang ini adalah Ron Rivest, Adi Shamir, dan Leonard Adleman Mereka kemudian

mempublikasikan temuan mereka pada tahun 1978 dan algoritma kriptografi kunci asimetri yang mereka temukan dikenal dengan nama algoritma kriptografi RSA. RSA itu sendiri merupakan akronim dari nama keluarga mereka, Rivest, Shamir dan Adleman.

Pada tahun 1983, Massachussets Institute of Technology menerima hak paten atas sebuah makalah berjudul *"Chryptographv Communication System and Method'* Yang mengaplikasikan penggunaan algoritma kriptografi RSA. (Yusuf Kurniawan, 2004).

II.5.1 Proses Enkripsi dan Deskripsi RSA

Secara umum ada beberapa besaran-besaran yang harus diperhatikan dalam algoritma RSA, yaitu:

1. p dan q adalah bilangan prima (rahasia)
2. n = p . q (tidak rahasia)
3. $\phi(n)=(p-1)(q-1)$ (rahasia)
4. e (kunci enkripsi) (tidak rahasia)
5. d(kunci deskripsi) (rahasia.)
6. m (p l a i n t e k s) (rahasia)
7. c(chipher teks) (tidak rahasia)

II.5.2 Algoritma Membangkitkan Pasangan Kunci

1. Memilih dua buah bilangan prima yaitu diberi simbol p dan q
2. Menghitung nilai n = p.q (n≠t- p, karena jika n = p, maka nilai n = p^2 dan akan dapat mudah mendapatkan nilai n).
3. Hitung ϕ (n) = (p-1)(q-1)
4. Memilih kunci publik e yang relatif prima terhadap ϕ(n).

5. Bangkitkan kunci private dengan menggunakan persamaan:

$$d = \frac{1 + k\Phi(n)}{e}$$

Hasil dari algoritma di atas adalah:

a. Kunci publik adalah pasangan (e,n)

b. Kunci private adalah pasangan (d,n)

II.5.3 Keamanan RSA

Pada dasarnya keamanan algoritma RSA ini terletak yaitu pada sulitnya memfaktorkan bilangan besar menjadi faktor-faktor prima. Pada RSA masalah pemfaktoran berbunyi : Faktorkan n menjadi dua faktor prima, p dan q. sedemikian hingga n = p.q. sekali u dapat difaktorkan menjadi p dan q, maka $\phi(n) = (p-1)(q-1)$ dapat dihitung.

II.6 Algoritma Kriptografi Rijndael

Algoritma kriptografi bernama Rijndael yang didesain oleh oleh Vincent Rijmen dan John Daemen asal Belgia keluar sebagai pemenang kontes algoritma kriptografi pengganti DES yang diadakan oleh NIST (National Institutes of Standards and Technology) milik pemerintah Amerika Serikat pada 26 November 2001. Algoritma Rijndael inilah yang kemudian dikenal dengan Advanced Encryption Standard (AES).

Setelah mengalami beberapa proses standardisasi oleh NIST, Rijndael kemudian diadopsi menjadi standard algoritma kriptografi secara resmi pada 22 Mei 2002. Pada 2006, AES merupakan salah satu algoritma terpopuler yang digunakan dalam kriptografi kunci simetrik. (Rifki Sadikin; 2012: 151)

AES ini merupakan algoritma block cipher dengan menggunakan sistem permutasi dan substitusi (P-Box dan S-Box) bukan dengan jaringan Feistel sebagaimana block cipher pada umumnya. Jenis AES terbagi 3, yaitu :

1.AES-128

2.AES-192

3.AES-256

Pengelompokkan jenis AES ini adalah berdasarkan panjang kunci yang digunakan. Angka-angka di belakang kata AES menggambarkan panjang kunci yang digunakan pada tipa-tiap AES. Selain itu, hal yang membedakan dari masing-masing AES ini adalah banyaknya round yang dipakai. AES-128 menggunakan 10 round, AES-192 sebanyak 12 round, dan AES-256 sebanyak 14 round.

AES memiliki ukuran block yang tetap sepanjang 128 bit dan ukuran kunci sepanjang 128, 192, atau 256 bit. Tidak seperti Rijndael yang block dan kuncinya dapat berukuran kelipatan 32 bit dengan ukuran minimum 128 bit dan maksimum 256 bit.

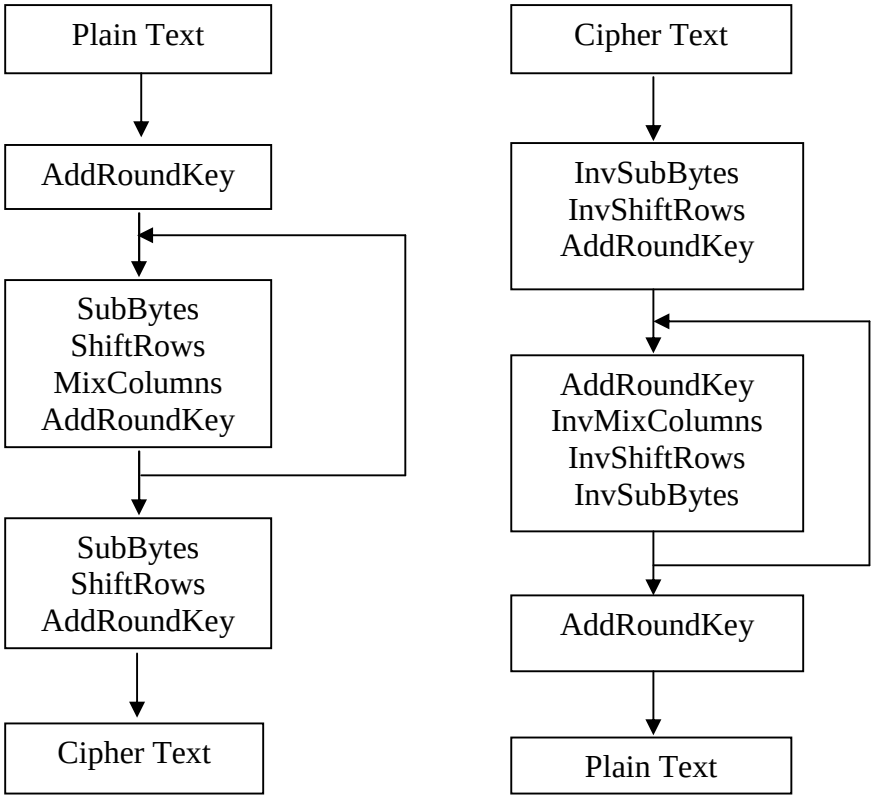
Berdasarkan ukuran block yang tetap, AES bekerja pada matriks berukuran 4x4 di mana tiap-tiap sel matriks terdiri atas 1 byte (8 bit). Sedangkan Rijndael sendiri dapat mempunyai ukuran matriks yang lebih dari itu dengan menambahkan kolom sebanyak yang diperlukan.

Algoritma Rijndael mempunyai 3 (tiga) parameter:

1. Plainteks adalah array yang berukuran 16-byte, yang berisi data masukan.
2. Cipherteks adalah array yang berukuran 16-byte, yang berisi hasil enkripsi.

3. Kunci adalah array yang berukuran 16-byte, yang berisi kunci cipher (disebut juga chiper key).
5. AddRoundKey, melakukan XOR antara state awal (plainteks) dengan chiper key. Tahap ini disebut juga initial round.
6. Putaran sebanyak Nr-1 kali. Proses yang dilakukan pada setiap putaran adalah
    a. SubBytes adalah substitusi byte dengan menggunakan tabel substitusi (S-Box).
    b. ShiftRows adalah pergeseran baris-baris array state secara wrapping.
    c. MixColumns adalah mengacak data di masing-masing kolom array state.
    d. AddRoundKey adalah melakukan XOR antara state sekarang round key.
7. Final round, proses untuk putaran terakhir:
    a. SubBytes
    b. ShiftRows
    c. AddRoundKey

Garis besar diagram algoritma Rijndael diperlihatkan pada gambar II.8 di bawah ini.



Gambar II.8 Diagram proses enkripsi dan proses dekripsi

II.6.1 Transformasi SubBytes

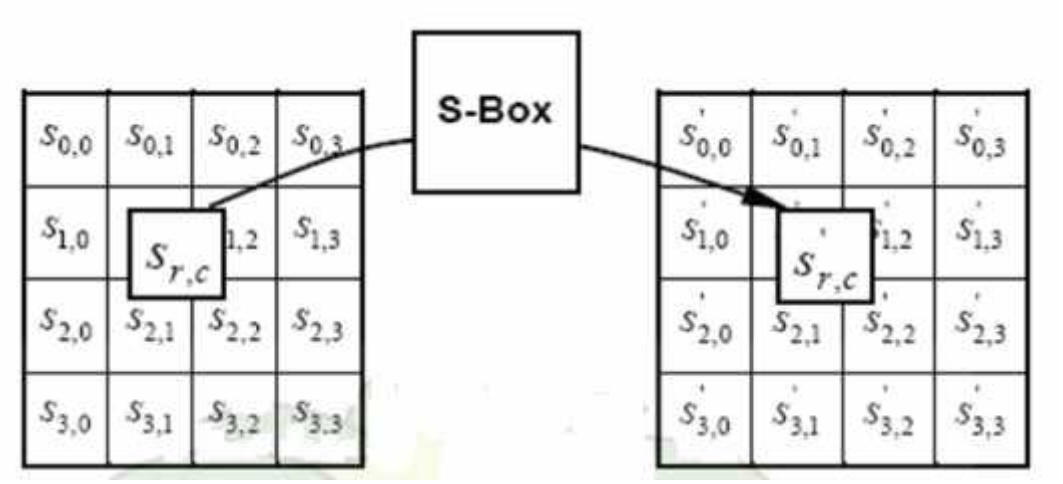
Transformasi SubBytes memetakan setiap byte dari array state dengan menggunakan tabel substitusi S-Box. Tidak seperti DES yang mempunyai S-Box berbeda pada setiap putaran, Rijndael hanya mempunyai satu buah. Tabel yang digunakan adalah:

Tabel II.9 S-Box Rijndael

hex	y															
	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

x

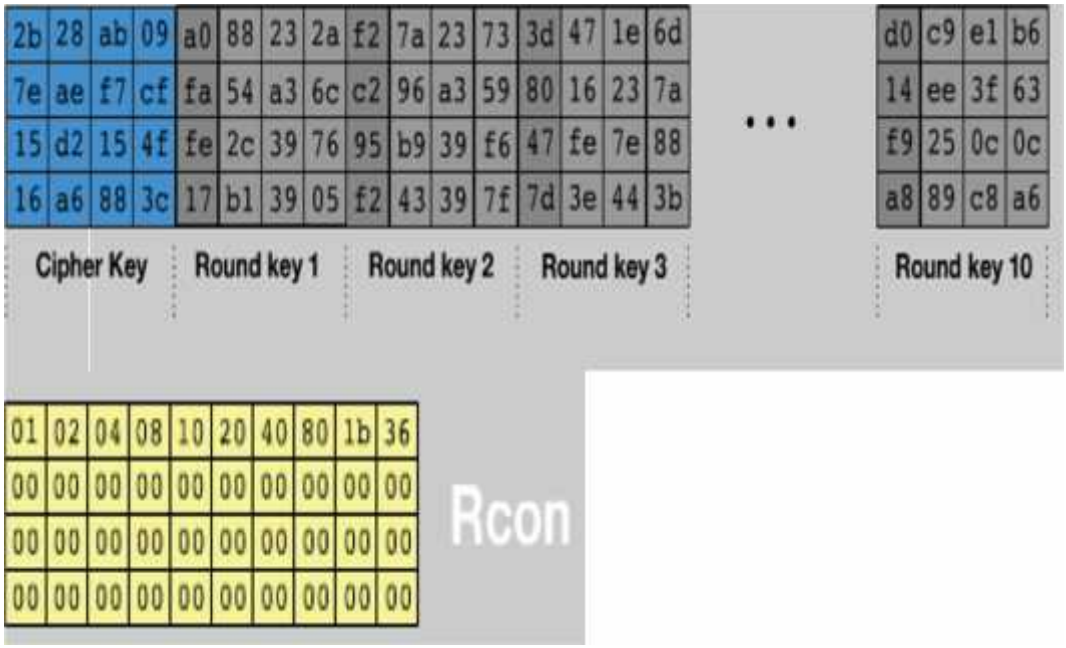
Cara pensubstitusian adalah sebagai berikut: jika setiap byte pada array state S[r,c]=xy, xy adalah digit heksadesimal dari nilai S[r,c], maka nilai substitusinya, dinyatakan dengan S'[r,c], adalah elemen di dalam S-Box yang merupakan perpotongan baris x dengan kolom y.



Gambar II.9 Transformasi SubByte dengan S-Box

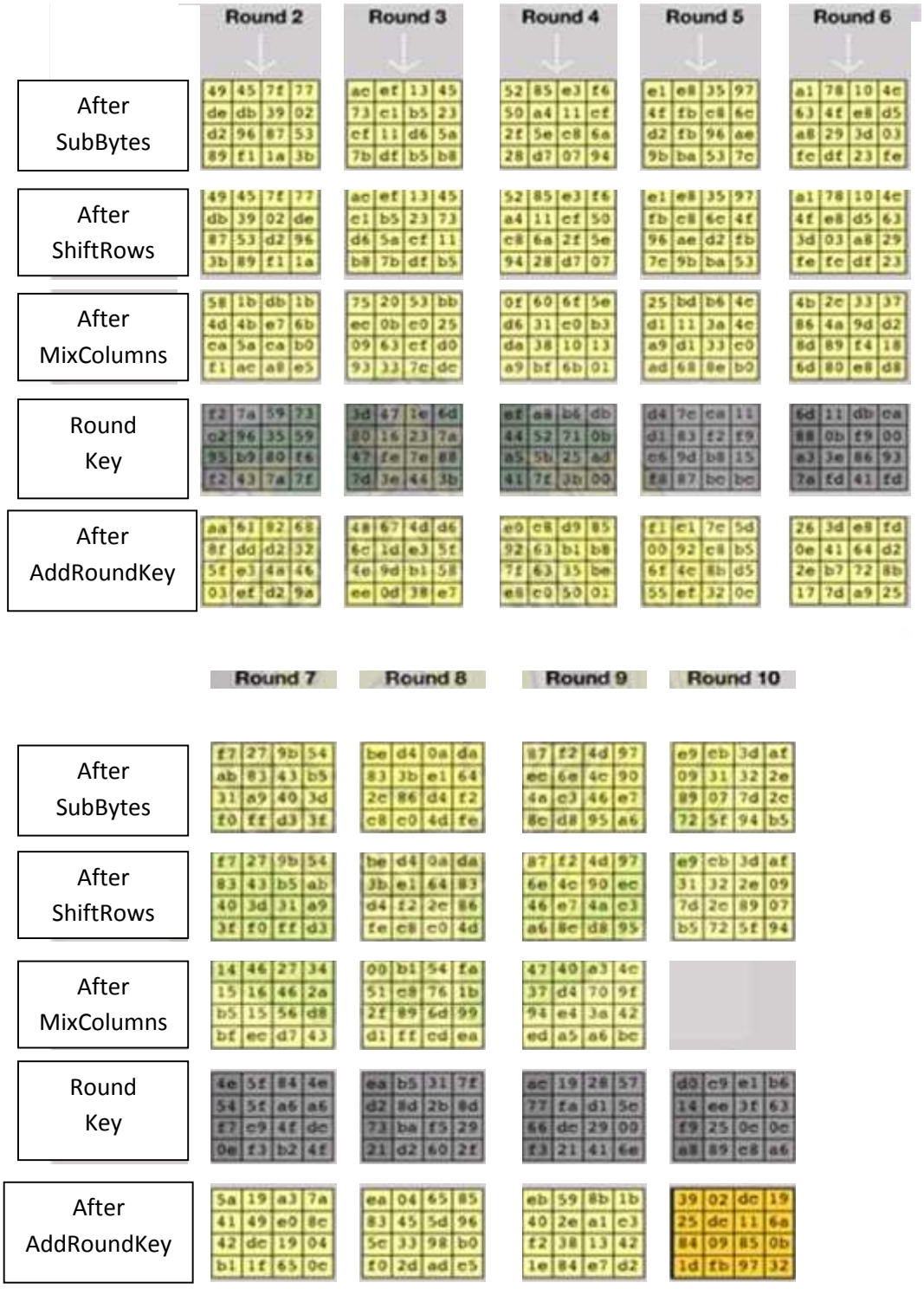
II.6.2 Ekspansi Kunci

Algoritma Rijndael melaksanakan cipher key dan membuat suatu ekspansi kunci untuk menghasilkan suatu key schedule. Jika ekspansi kunci yang diperlukan Rijndael Nb(Nr+1) word, sehingga bisa digunakan AES 128 bit, maka 4(10+1)=40 word=44x32 bit=1408 bit subkey. Ekspansi dari 128 menjadi 1408 bit subkey, proses ini disebut dengan key schedule. Subkey ini diperlukan karena setiap round merupakan suatu inisial dari Nb word untuk Nr=0 dan Nb untuk Nr=1,3 untuk Nr= 2, 11 Nb untuk Nr=10, dari operasi ini akan didapatkan schedule kunci yang berisi array linier 4 byte word (w_i), 0=i(Nr+1).



Gambar II.10 Proses ekspansi kunci

Contoh dari keseluruhan proses Algoritma Rijndael



Gambar II.11 Proses algoritma Rijndael

II.7 Keamanan Rijndael

Untuk Rijndael, tipe serangan square attacks cukup menjadi dikenal sebagai serangan terbaik terhadap Rijndael. Square attacks adalah serangan yang memanfaatkan struktur orientasi byte. Algoritma ini bekerja dengan baik pada square cipher yang bekerja dalam 6 putaran.

Apabila Rijndael dengan kunci sepanjang 128 bit, maka serangan ini lebih cepat dari pada exhaustive search hingga 6 kali iterasi Rijndael. Namun, untuk AES jelas bahwa serangan ini tidak mungkin dipraktekkan karena jumlah putaran pada Rijndael, mengakibatkan batas keamanan untuk algoritma ini menjadi lebih besar. Pada tahun 2002 melalui suatu proses pengujian yang sifatnya teoritis ditemukan bahwa AES mungkin dapat dijebol atau dipecahkan. Metode attack ini dinamakan "XLS Attack".

Serangan ini pertama kali dipublikasikan oleh Nicolas Courtois dan Josep Pieprzyk dalam makalah mereka yang berjudul "Cryptanalysis of Block Ciphers with Overdefined Systems of Equations". Teknik ini diklaim dapat memecahkan AES lebih cepat dari cara exhaustive search. XSL attack mengandalkan pada keberhasilan menganalisis subsistem internal dari cipher untuk menurunkan persamaan kuadrat secara simultan. Kumpulan persamaan ini umumnya sangat besar. Contohnya pada 128 bit AES terdapat 8000 persamaan dengan jumlah variabel 1600. Metode untuk memecahkan persamaan ini disebut XSL (eXtended Sparse Linearisation). Jika persamaan tersebut dapat dipecahkan, maka kunci dapat diperoleh.

Jika Pemecahan persamaan tersebut menjadi masalah, maka ditemukan persamaan yang bersifat MQ(Multivariate quadratic). Persamaan MQ merupakan permasalahan yang bersifat NP-hard (Non Polinomial). XSL attack membutuhkan algoritma yang efisien untuk menyelesaikan MQ. Salah satu teknik untuk menyelesaikan sistem MQ adalah dengan linearisasi, yang mengubah setiap persamaan kuadrat menjadi variabel yang independen yang akan menghasilkan persamaan linear dengan menggunakan algoritma seperti Gaussian elimination.

Tahun 2000, Courtois mengajukan algoritma untuk MQ yang bernama XL(eXtended Linearisastion). Algoritma ini meningkatkan jumlah persamaan dengan mengalikan dengan monomial derajat tertentu. Algoritma ini akan menghasilkan suatu bentuk struktur yang disebut XSL. Algoritma XSL dibentuk dari algoritma XL dengan memilih monomial secara selektif.

II.8 Visual Basic 2008

Program Visual Basic 2008 adalah versi terbaru program Visual Basic saat buku ini dibuat. Seperti yang kita ketahui, program Visual Basic adalah bahasa pemograman yang paling mudah di kuasai oleh para pemula. Dalam versi terbaru ini, program Visual Basic 2008 (disingkat VB 2008) menawarkan banyak kemudahan lagi dibandingkan versi versi sebelumnya, antara lain teknik pemograman dapat dibuat lebih terstruktur dan lebih banyak bantuan dalam pemograman. Jauh lebih mudah untuk menguasai dibandingkan dengan versi terdahulunya, yaitu Visual Basic 6 (disingkat VB 6).

Ada banyak perubahan dalam VB 2008 ini dibandingkan VB 6 , antara lain:

1. Bahasa pemograman sekarang benar benar bahasa berbasis objek (Object Oriented Programing, sedangkan VB 6 bukan bahasa berbasis object.
2. Aplikasi dan komponen yang ditulis di VB 2008 mempunyai akses penuh ke Net FrameWork. Sedangkan di VB 6 tidak dikenal atau tidak digunakan Net FrameWork.
3. Semua aplikasi yang dibuat beroperasi dalam manajement *Common Languages Runtime* (CLR).

II.9 Cryptografi Net Framework

Cryptografi net Framework adalah salah satu modul yang diterapkan pada *Microsoft Visual Basic 2005*. Dimana *Cryptografi net Framework* adalah berfungsi sebagai alat untuk membantu enkripsi dan dekripsi. Di dalam *Cryptografi net Framework* terdapat *CryptoStream* yaitu salah satu class yang banyak di sediakan. Kelas*CryptoStream* dirancang untuk mengenkripsi atau mendekripsi konten seperti itu mengalir keluar ke file. Pada perancangan aplikasi teks editor ini, penulis menggunakan Cryptografi Net Framework sebagai media perancangan metode enkripsi.