SBS : Simple But Secure Block Cipher

Sebuah *block cipher* yang sederhana namun aman dengan ukuran blok dan jumlah putaran berdasarkan kunci pengguna.

Kristianto Karim
Teknik Informatika / Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Bandung, Indonesia
13514075@std.stei.itb.ac.id

Chalvin
Teknik Informatika / Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Bandung, Indonesia
13514032@std.stei.itb.ac.id

***Abstrak*—Block Cipher merupakan algoritma yang beroperasi pada sekumpulan bit dengan panjang tertentu yang disebut dengan *block*. Makalah ini mengajukan sebuah *block cipher* baru yang sederhana namun memiliki tingkat keamanan yang cukup tinggi. SBS merupakan *block cipher* yang menerapkan jaringan feistel untuk enkripsi dan dekripsinya. SBS juga menerapkan prinsip *diffusion* dan *confusion* dari Shannon. Algoritma ini juga menerapkan jumlah perputaran dan ukuran blok serta ukuran kunci yang berbeda yang bergantung terhadap kunci yang diberikan oleh pengguna. SBS juga dapat beroperasi pada berbagai mode operasi seperti ECB, CBC, CFB, OFB, dan CTR.**

Kata kunci: *block, cipher, encrypt, decrypt, block cipher, algorithm, secure, feistel, shannon, diffusion, confusion, round*, enkripsi, dekripsi, algoritma

I. PENDAHULUAN

Kriptografi berasal dari bahasa Yunani *kryptós* dan *graphein*. *Kryptós* artinya menyembunyikan dan *graphein* artinya tulisan sehingga arti kriptografi adalah ilmu yang mempelajari cara-cara atau teknik-teknik yang digunakan untuk menjaga keamanan informasi, kerahasiaan data, keabsahan data dan integritas serta otentikasi data dalam komunikasi yang aman dengan cara mengonversi pesan menjadi bentuk yang tidak bermakna bagi orang yang tidak berwenang.

Penggunaan kriptografi awal yang diketahui yaitu berupa teks sandi yang terukir pada batu di Mesir pada tahun 1900 sebelum Masehi. Kemudian, kriptografi berkembang terus-menerus dan banyak digunakan terutama dalam bidang militer. Dahulu, enkripsi yang dilakukan hanya memanfaatkan transposisi atau substitusi. Saat ini, muncul berbagai teknik yang membuat kriptografi menjadi lebih kompleks. Seiring berkembangnya zaman, semakin banyak juga penerapan ilmu kriptografi dalam kehidupan sehari-hari terutama pada internet dimana pertukaran data terjadi secara terus-menerus. Hal ini mendorong dikembangkannya algoritma kriptografi yang lebih aman. Berbagai prinsip untuk meningkatkan tingkat keamanan, seperti prinsip permutasi, substitusi, *confussion* dan *diffusion*, kemudian menjadi prinsip-prinsip dasar yang perlu diterapkan. Berbagai algoritma pun telah dikembangkan dan digunakan secara massal, salah satunya yaitu kriptografi *block cipher*.

Pada makalah ini, akan dibahas rancangan *block cipher* baru yang memiliki proses enkripsi dan dekripsi yang sederhana namun menghasilkan *ciphertext* yang kompleks sehingga sulit dipecahkan.

II. DASAR TEORI

Pada bab ini akan dibahas dasar-dasar teori yang digunakan sebagai referensi dalam pembangunan algoritma yang diajukan pada makalah ini.

A. *Block Cipher*

Block cipher adalah sebuah fungsi yang memetakan n-bit *plaintext* menjadi n-bit *ciphertext* dengan n adalah panjang blok (Menezes, dkk., 1996). Pesan yang dienkripsi akan dipecah menjadi bagian kecil seukuran panjang blok n lalu blok-blok tersebut dijadikan masukan dari fungsi *block cipher*.

Block cipher umumnya beroperasi pada beberapa mode seperti *Electronic Codebook* (ECB), *Cipher Block Chaining* (CBC), *Cipher Feedback* (CFB), *Output Feedback* (OFB), dan *Counter* (CTR). Berikut adalah penjelasan mengenai setiap mode pada *block cipher*.

1) *Electronic Codebook* (ECB)

Mode ECB merupakan mode yang paling sederhana dibandingkan dengan mode-mode yang lain. Pada mode ini, pesan awalnya dipecah terlebih dahulu menjadi blok-blok dengan panjang yang sama. Pada proses enkripsi, masing-masing blok dienkripsi dengan fungsi enkripsi pada *blok cipher*. Hasil enkripsi tersebut kemudian digabungkan untuk menghasilkan *ciphertext*. Pada proses dekripsi, *ciphertext* dibagi terlebih dahulu ke dalam blok-blok dengan panjang yang sama dengan panjang blok saat enkripsi. Kemudian, tiap-tiap blok menjadi masukan invers dari fungsi enkripsi. Mode ini sudah jarang dipakai karena dapat memberikan informasi pola pada data *plaintext*.

2) *Cipher Block Chaining* (CBC)

Mode ini mirip dengan mode ECB. Perbedaannya, mode ini melakukan modifikasi *plaintext* terlebih dahulu sebelum menjadi masukan dari fungsi enkripsi. Modifikasi ini berupa operasi XOR *plaintext* pada blok ke-n dengan *ciphertext* yang didapatkan dari blok ke-(n-1). Pada blok pertama, *plaintext* di-XOR dengan suatu *Initialization Vector* (IV). Hal ini membuat pola dari *plaintext* lebih sukar untuk dideteksi dari *ciphertext*. Proses dekripsi dari mode

ini adalah sebagai berikut: blok cipherteks didekripsi seperti biasa, kemudian hasil dekripsi tersebut di-XOR-kan dengan cipherteks pada blok sebelumnya. Untuk blok pertama, hasil dekripsi di-XOR dengan *initialization vector* yang dipakai pada proses enkripsi. Perlu diperhatikan bahwa *initialization vector* tidak perlu dirahasiakan dari orang lain.

Karena proses enkripsi pada mode ini bergantung dari hasil fungsi enkripsi pada blok sebelumnya, proses enrkipsi pada mode ini tidak dapat diparelisasikan. Kesalahan pada *ciphertext* ke-n akan berpengaruh pada hasil dekripsi pada blok ke-n dan (n+1). Perubahan 1 bit pada blok *ciphertext* ke-n akan menyebabkan hasil dekripsi pada blok ke-n memiliki kesalahan sekitar 50%. Blok ke-(n+1) akan memiliki kesalahan bit pada posisi yang sama dengan kesalahan pada blok *ciphertext* ke-n.

3) *Cipher Feedback* (CFB)

Berbeda dengan mode sebelumnya, mode CFB tidak menggunakan blok *plaintext* sebagai masukan dari fungsi enkripsi. Proses enkripsi dilakukan dengan mengenkripsi blok *ciphertext* pada blok sebelumnya. Hasilnya kemudian di-XOR dengan *plaintext* untuk menghasilkan *ciphertext*. Pada blok pertama, yang dienkripsi bukanlah *ciphertext* pada blok sebelumnya melainkan *initialization vector*. Pada proses dekripsi, *ciphertext* pada blok sebelumnya dienkripsi (bukan didekripsi). Hasil enkripsi tersebut kemudian di-XOR dengan *ciphertext* pada blok yang sedang diproses untuk mendapatkan *plaintext*. Pada blok pertama, *initialization vector* menggantikan posisi *ciphertext* sebagai blok yang dienkripsi.

Pada mode CFB, pada proses enkripsi dan dekripsi keduanya memakai fungsi enkripsi. Karenanya, fungsi yang dipakai tidak perlu memiliki invers. Kelemahannya sama dengan kelemahan pada CBC.

4) *Output Feedback* (OFB)

Mode ini sangat mirip dengan mode CFB. Pada CFB, *ciphertext* pada blok sebelumnya dijadikan masukan dari operasi enkripsi blok. Hasil enkripsi tersebut kemudian di-XOR dengan *plaintext* untuk mendapatkan *ciphertext*. *Ciphertext* ini kemudian dijadikan masukan dari operasi enkripsi blok pada blok berikutnya. Pada OFB, yang menjadi masukan adalah *ciphertext* hasil enkripsi blok sebelumnya yang belum di-XOR dengan *plaintext*.

Proses enkripsi dan dekripsi pada OFB tidak bergantung pada blok sebelumnya. Jika terjadi kesalahan pada blok ke-n pada *ciphertext*, kesalahan hanya akan berdampak pada blok ke-n.

5) *Counter* (CTR)

Mode CTR menggunakan gabungan *nonce* dan suatu *counter* sebagai masukan dari fungsi enkripsi blok. Hasil enkripsi tersebut kemudian di-XOR dengan *plaintext* untuk mendapatkan *ciphertext*. Untuk tiap-tiap blok, *nonce* yang dipakai sama namun dengan *counter* yang berbeda. Umumnya untuk setiap blok, *counter* akan bertambah satu.

Dimana g(i) umumnya merupakan fungsi identitas. Proses enkripsi dan dekripsi pada mode ini tidak bergantung pada blok-blok sebelumnya sehingga proses enkripsi dan dekripsi dapat dipararelisasi.

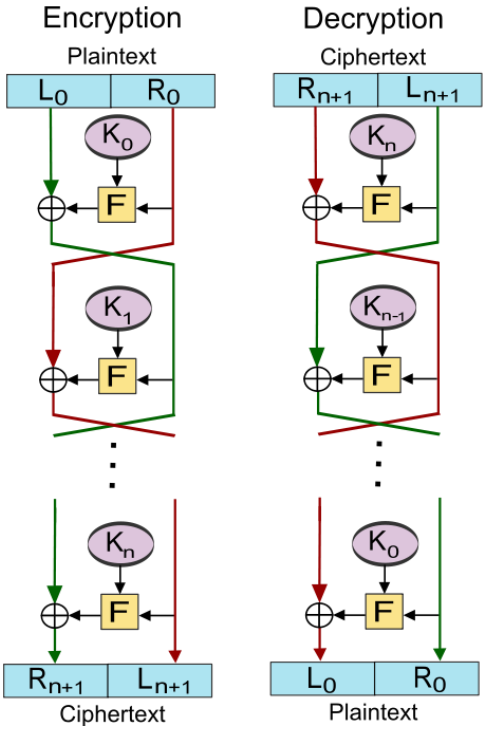
B. *Feistel Cipher*

Feistel cipher atau jaringan feistel adalah sebuah struktur simetris yang digunakan dalam pembangunan *block cipher*. *Feistel cipher* ini diperkenalkan pertama kali oleh kriptografer bernama Horst Feistel. Cukup banyak *block cipher* menggunakan struktur ini. Salah satu yang paling terkenal adalah *Data Encryption Standard* (DES). Banyak digunakannya jaringan feistel dikarenakan operasi untuk enkripsi dan dekripsi sangatlah mirip bahkan dalam beberapa kasus operasi yang dilakukan sama persis.

Jaringan feistel pada algoritma ini secara garis besar akan melakukan serangkaian operasi berikut.

1. Membagi blok menjadi dua bagian yaitu bagian kiri dan kanan.
2. Memproses blok kanan dengan memasukkannya ke dalam fungsi putaran dengan kunci putaran yang dibangkitkan.
3. Blok bagian kiri kemudian di-XOR dengan hasil fungsi putaran.
4. Hasil fungsi putaran kemudian di gabungkan dengan hasil XOR pada tahap 3.
5. Tahap 1 sampai 4 dilakukan sebanyak jumlah putaran yang ditentukan.

Skema dari jaringan feistel pada algoritma yang diajukan dapat dilihat pada Gambar 2-1.



Gambar 2-1. Proses enkripsi dan dekripsi dengan *feistel cipher*. Sumber : https://en.wikipedia.org/wiki/File:Feistel_cipher_diagram_en.svg

C. *Confusion* dan *Diffusion*

Confusion dan *Diffusion* merupakan suatu properti dalam pembangunan teks terenkripsi untuk meningkatkan keamanan yang diperkenalkan oleh Claude Elwood Shannon pada tahun 1949. Berikut adalah penjelasan dari *confusion* dan *diffusion*.

1) *Confusion*

Prinsip *confusion* merupakan suatu prinsip untuk membuat hubungan antara kunci dan *ciphertext* serumit mungkin. Hal tersebut dapat dicapai dengan membuat setiap bagian atau karakter dari *ciphertext* yang dihasilkan bergantung terhadap beberapa bagian kunci. Tujuannya agar hubungan antara kunci dengan *ciphertext* dapat disembunyikan.

2) *Diffusion*

Prinsip *diffusion* merupakan suatu prinsip dimana perubahan suatu karakter pada *plaintext* akan mempengaruhi perubahan *ciphertext* yang dihasilkan secara signifikan. Perubahan satu karakter diharapkan

mengakibatkan setengah bahkan seluruh *ciphertext* yang dihasilkan berbeda.

III. RANCANGAN ALGORITMA

Algoritma yang diajukan pada makalah ini menggunakan jaringan feistel dengan ukuran blok X, jumlah putaran N dan ukuran key K yang bergantung terhadap kunci yang dimasukkan pengguna. Setelah nilai N, X dan K ditentukan *plaintext* akan dibagi berdasarkan ukuran blok. Kunci yang digunakan juga akan dipraproses agar panjangnya mencapai K. Setelah itu setiap blok dan kunci yang telah dipraproses tersebut akan diproses melalui jaringan feistel. Berikut adalah penjelasan untuk masing-masing fitur pada algoritma yang diajukan.

A. Penentuan Ukuran Blok, Ukuran Kunci dan Jumlah Putaran

Ukuran blok, kunci dan jumlah putaran ditentukan oleh kunci yang dimasukkan oleh pengguna. Berikut adalah penjelasan cara penentuan ukuran blok, kunci dan jumlah putaran berdasarkan kunci pada algoritma yang diajukan.

1) Ukuran blok
Ukuran blok pada algoritma ini adalah kelipatan 64 bit dengan minimal adalah 64 bit (64 * 1) dan maksimalnya 264 (4 * 64) bit. Penentuan angka kelipatan dilakukan dengan rumus berikut.

$$X = (\text{Panjang Kunci} \ \% \ 4) + 1$$

Contoh :

Kunci dari pengguna : KunciSuperAman12345!?

Panjang Kunci : 21

Kelipatanya : (21 % 4) + 1 = 2

Panjang blok : Kelipatan * 64 bit = 2 * 64 = 128 bit

Sehingga panjang blok yang digunakan untuk proses enkripsi maupun dekripsi dengan panjang kunci tersebut adalah 128 bit.

2) Ukuran kunci
Ukuran kunci pada algoritma ini ditentukan dengan melakukan penjumlahan nilai ordinal dari setiap karakter dari kunci yang diberikan oleh pengguna–kemudian di bagi dengan panjang kunci dan dibulatkan ke bawah. Jika panjang kunci pengguna tidak mencapai ukuran kunci maka kunci baru akan dibangkitkan. Pembangkitan kunci dilakukan dengan melakukan perulangan kunci dari pengguna dan kunci dari pengguna yang dibalikkan urutannya secara selang seling.
Contoh.
Kunci dari pengguna : ABCDEFGHIJK
Ukuran kunci : (65 + 66 + 67 + 68 + 69 + 70 + 71 + 72 + 73 + 74 + 75) / 411 = 70
Kunci yang dibangkitkan :
ABCDEFGHIJKKJIHGFEDCBAABCDEFGHIJKKJI
HGFEDCBAABCDEFGHIJKKJIHGFEDCBAABCD
Maka kunci yang dibangkitkan akan digunakan sebagai kunci enkripsi atau dekripsi yang dilakukan dalam SBS.

3) Jumlah Putaran
Jumlah putaran pada algoritma ini ditentukan oleh panjang kunci yang di berikan pengguna. Namun jika panjang kunci yang diberikan kurang dari 12, panjang kunci tidak akan digunakan sebagai jumlah putaran dan jumlah putaran yang digunakan adalah 12.
Contoh :
1. Kunci : KunciSuperAman12345!?
   Panjang Kunci : 21
   Jumlah putaran : 21
2. Kunci : ABCD
   Panjang Kunci : 4
   Jumlah putaran : 12

B. Pembangkit Kunci Putaran

Kunci putaran pada algoritma blok cipher ini dibangkitkan dengan cara berikut. Pada putaran ke-n, kunci dirotasi sebanyak n ke kiri. Kemudian akan mengubahnya menjadi matriks 8xN dan melakukan pencerminan secara horizontal lalu vertikal. Setelah itu, kunci akan dibagi menjadi dua bagian yaitu bit kunci pada posisi genap akan dikumpulkan menjadi satu bagian, sedangkan bit kunci pada posisi ganjil akan dikumpulkan menjadi bagian lainnya(bagian kanan). Setengah bagian pertama dari kunci akan di isi dengan bagian genap lalu kunci akan dinegasikan. Setengah bagian sisanya kemudian akan diisi dengan bagian ganjil lalu dinegasikan.

Berikut adalah contoh pembangkitan kunci untuk putaran ke-4.

Kunci Eksternal :
10010111 11010111 01110110 00111101
00110100 11111010 11110000 11001011

Pembangkitan Kunci Putaran :
1. Hasil rotasi sebanyak 4 kali ke kiri
01111101 01110111 01100011 11010011
01001111 10101111 00001100 10111001
2. Mengubah kunci ke bentuk matriks dan melakukan pencerminan secara horizontal lalu secara vertikal

Hasil perubahan ke bentuk matriks	Hasil pencerminan secara horizontal	Hasil pencerminan secara vertikal
01111101	10111110	10011101
01110111	11101110	00110000
01100011	11000110	11110101
11010011	11001011	11110010
01001111	11110010	11001011
10101111	11110101	11000110
00001100	00110000	11101110
10111001	10011101	10111110

Hasil setelah pencerminan :
10011101 00110000 11110101 11110010
11001011 11000110 11101110 10111110
3. Membagi kunci menjadi dua bagian yang terdiri dari bit ganjil dan bit genap
Ganjil : 10100100 11001101 10111001 11111111
Genap : 01110100 11111100 10011010 10100110
4. Mengisi setengah bagian pertama kunci hasil pencerminan dengan bit kunci genap dan menegasikannya.
Hasil pengisian sebagian pertama :
01110100 11111100 10011010 10100110
11001011 11000110 11101110 10111110
Hasil negasi :
10001011 00000011 01100101 01011001
00110100 00111001 00010001 01000001
Hasil pengisian sebagian terakhir :

```
10001011 00000011 01100101 01011001
10100100 11001101 10111001 11111111
Hasil negasi :
01110100 11111100 10011010 10100110
01011011 00110010 01000110 00000000
Kunci putaran ke-4 yang digunakan adalah
01110100 11111100 10011010 10100110
01011011 00110010 01000110 00000000
```

C. Fungsi Putaran

Secara umum, fungsi putaran digunakan dalam jaringan feistel. Jaringan feistel akan membagi blok menjadi bagian kiri dan kanan. Blok bagian kanan dan kunci putaran akan menjadi masukan dari fungsi putaran. Fungsi putaran akan memproses blok bagian kanan melalui serangkaian operasi sebagai berikut.

1. Mengubah blok bagian kanan tersebut menjadi matriks berukuran 8xN. Dimana N adalah ½ ukuran blok dibagi 8. Setelah itu, matriks akan di cerminkan secara horizontal. Kemudian matriks akan dicerminkan secara vertikal. Lalu, matriks akan diubah kembali menjadi array 1 dimensi.
2. Setelah itu, akan dilakukan S-Box substitution pada blok. SBS melakukan S-Box substitusion sebanyak 8 kali. Untuk S-Box ke-n, bit ke-n*6 sampai ke-n*6 + 6 dari blok yang sedang diproses diambil dan disubstitusi berdasarkan tabel S-Box ke-n. S-Box yang digunakan dihasilkan secara pseu-random dan dapat dilihat pada lampiran.
3. Blok bagian kanan yang telah disubstitusi kemudian di-XOR dengan kunci putaran.
4. Setelah itu, blok bagian kanan akan dirotasi ke kiri sebanyak 3 bit.
5. Kemudian blok bagian kanan akan di-XOR sekali lagi dengan kunci putaran.
6. Selanjutnya akan dilakukan pertukaran posisi setiap dua byte pada blok bagian kanan.
7. Terakhir blok bagian kanan akan di-XOR lagi dengan kunci putaran.

Skema dari langkah-langkah yang dilakukan pada fungsi putaran dapat dilihat pada Gambar 3-1.

Gambar 3-1. Operasi-operasi yang terjadi di dalam fungsi putaran.

IV. PENGUJIAN

Pada bab ini akan dijelaskan mengenai pengujian yang telah dilakukan dengan SBS. Berikut adalah beberapa hasil pengujian yang telah dilakukan dengan *block cipher* yang diajukan makalah ini.

A. Pengujian Enkripsi dengan Berbagai Mode Operasi

Pada subbab ini akan dipaparkan hasil pengujian SBS dengan berbagai mode operasi.

Tabel 4-1 Hasil pengujian mode CBC

Plain	acer toshiba asus lenovo mac del del xps 15 13
Kunci	h4kunam47474
Cipher	• P□?…®SþVeáè€Û±MþY…8– ®FâP~¨æ„•¿Hä È5□àMñS~¡â‹•£]û □eÜö*f*"

Tabel 4-2 Hasil pengujian mode ECB

Plain	acer toshiba asus lenovo mac del del xps 15 13
Kunci	1234
Cipher	/dBö6„IŸ/q^ð?ŽNŒa□[ê$Ç@ˆ/tHéwŠMŽ/h]öw *f*I□>#/‡qÖÍ

Tabel 4-3 Hasil pengujian mode CFB

Plain	ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
Kunci	h4kunam47474h0h0_huPl4
Cipher	)¹€ÿ————————————————$bé°J,#_ è9íÅçB"\|ãâ-Ï&Wê7¿Á«F2 ————————————————s¬â<†2Lât¿õ »F: ———————————————— t¨¯}Ã1LípÞõ¼V=-5«ì.Ä- [í{éãºG!tºý$ ‡1Cànïï»G8 ————————————————z´â.‡3Kñh¿ó ¼G7`¬·2‡7_,|«º¯?/-¹2‡4A²i±û¸IRcI

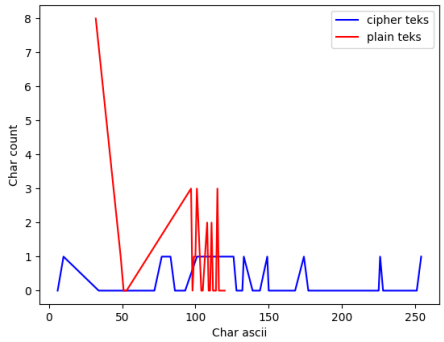
Tabel 4-4 Hasil pengujian mode OFB

Plain	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
Kunci	abcdefghijklmnopqrstuvwxyz
Cipher	~MSõC • ØÄ————————————————txf am7š/à4—Åmp p m0š- ü#ÔÂ————————————————m]c@t e=:† æ(Ð– ————————————————uAeL5 i)s',¯#ÞÃtGu@aa=<‡cæ(ÔßpLda^ KuóE‰

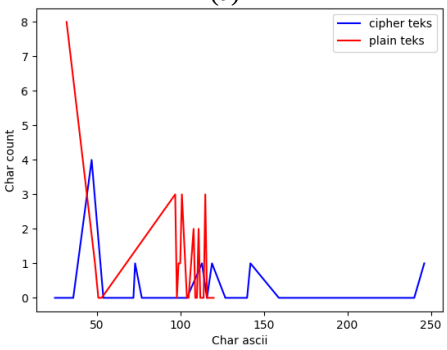
Tabel 4-5 Hasil pengujian mode CTR

Plain	Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt
Kunci	h4h4h4h4
Cipher	°P"bù×Ê Ì.H±ÆÍÕIÅaM®Ç˜Ë ————————————————Õ.@¬ÐÌ"K ÂaO²ÐÛÌ Õ{SáÔÜÑÈ}B¨Ûß˜ ÍgUí•ËÝ □jNáÐÑÍÌaEáÁÝÕÎ|¨ÛÛÑÈjT¯Á˜¾e§'Ç

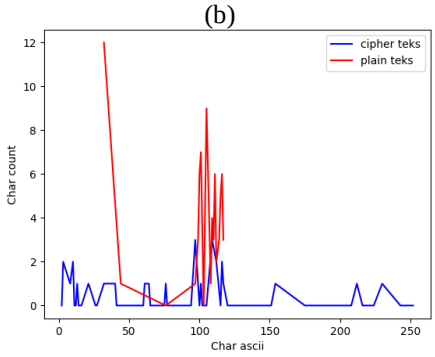
Grafik frekuensi kemunculan karakter dari enkripsi SBS dalam berbagai mode dapat dilihat pada Gambar 4-1.



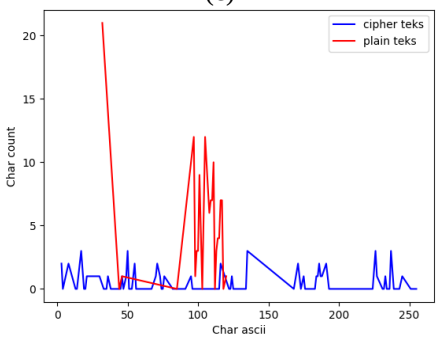
(a)



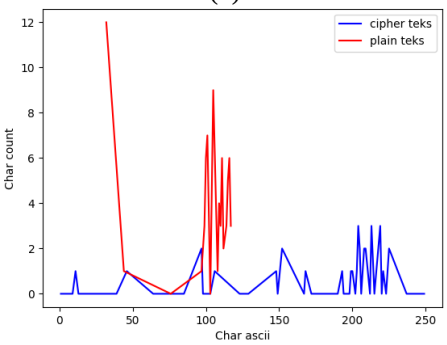
(b)



(c)



(d)



(e)

Gambar 4-1. Grafik Perbandingan Frekuensi Karakter antara *Plaintext* dan *Ciphertext*
(a) CBC (b) ECB (c) OFB (d) CFB (e) CTR

B. Pengujian Enkripsi dan Dekripsi dengan Perubahan Kunci

Pada subbab ini akan dipaparkan hasil pengujian enkripsi dan dekripsi dimana perubahan kecil pada kunci dilakukan. Pengujian dilakukan menggunakan mode CFB.

1) Enkripsi

Enkripsi dilakukan pada *plaintext* dengan dua buah kunci yang berbeda satu karakter.

Tabel 4-6 Hasil pengujian enkripsi dengan penggantian kunci

Plain	Uvuvwevwevweonyetenyetvwe 0s4s!
Kunci 1	Kunciseterongsatu
Kunci 2	Konciseterongsatu
Cipher 1	j±ÅU • ÔáÇ>-šyÒK-.:SjßR,ÿ›£†Ÿ,YÝ-öÓ/ž—~
Cipher 2	ÓŸiu/ • ÊýØof@üçøæXvÀ úß$öYyäsÖÁE Ø • !Â¡

2) Dekripsi

Dekripsi dilakukan pada *ciphertext* yang dihasilkan dengan mengenkripsi *plaintext* dengan Kunci 1.

Tabel 4-7 Hasil pengujian dekripsi dengan penggantian kunci

Plain	Uvuvwevwevweonyetenyetvwe 0s4s!
Kunci 1	Kunciseterongsatu
Kunci 2	Kunciseterongsstu
Dekripsi 1	Uvuvwevwevweonyetenyetvwe 0s4s!
Dekripsi 2	\ ühMýikl þ{Uöfy}çg_ìiklR¹më>

C. Pengujian Enkripsi dengan Perubahan *Plaintext*

Pada subbab ini akan dipaparkan hasil pengujian enkripsi dengan kunci dan IV yang sama terhadap dua buah *plaintext* yang berbeda satu buah karakter. Pengujian ini dilakukan dengan mode operasi OFB.

Tabel 4-8 Hasil pengujian enkripsi dengan penggantian *plaintext*

Plain 1	UvuvwevwevweonyetenyetveA 0s4s!
Plain 2	UvuvwevwevweonyetenyetveB 0s4s!
Kunci	Yo_gais_ini_kunci_seterong_123
Cipher 1	sÕ»Ø□žcp»□åz□~ÕhšQ□¯ ycþŽýM” @ • *¢ê‘ ˆ d
Cipher 2	sÕ»Ø□žcp»□åz□~ÕhšQ□¯ ycþŽýM” C • *¢ê‘ ˆ d

V. ANALISIS KEAMANAN

SBS menerapkan *confussion* dan *diffusion* sehingga serangan statistik menjadi lebih sulit dilakukan. Penerapan prinsip *confussion* dan *diffusion* dapat dilihat dari hasil pengujian pada Bab IV, dengan perubahan kunci dapat kita lihat bahwa hubungan kunci dengan cipher teks sangat kompleks. Hal tersebut terlihat dari saat melakukan dekripsi, perbedaan kecil pada kunci menyebabkan perbedaan yang jauh dari hasil dekripsi dengan file aslinya. Pengujian untuk enkripsi dengan kunci yang perbedaannya sangat kecil juga menghasilkan cipher teks yang sangat berbeda secara signifikan.

Begitu juga dengan pengujian untuk plain teks yang berbeda sedikit dengan kunci yang sama. Dari hasil pengujian tersebut, dapat kita lihat perubahan 1 buah karakter pada plain teks. menghasilkan cipher teks yang sangat berbeda.

Pembangkitan kunci internal atau kunci putaran pada SBS juga cukup rumit dimana operasi rotasi, pencerminan dan negasi dilakukan. Selain pembangkitan kunci yang rumit, fungsi putaran pada SBS juga tergolong kompleks karena melakukan pencerminan blok baik secara horizontal maupun vertikal, melakukan substitusi S-Box, melakukan XOR dengan kunci internal, menerapkan operasi rotasi dan *swapping* setiap dua *bytes*. Berikut akan dibahas serangan berupa analisis frekuensi dan serangan *bruteforce* pada SBS.

A. Analisis Frekuensi

Pada serangan analisis frekuensi, kriptanalis akan menganalisis kemunculan karakter pada cipher teks dan mencoba menebak plain teks berdasarkan frekuensi karakter. Berdasarkan hasil pengujian pada gambar 4-1 dapat diihat bahwa cipher teks yang dihasilkan terdiri dari karakter yang tersebar merata dan bervariasi meskipun plain teks dan kunci diubah. Hal ini membuat SBS memiliki tingkat keamanan yang cukup baik, karena kunci atau plainteks yang bagaimanapun akan menghasilkan cipher teks yang distribusi karakternya merata. Karena distribusi karakternya merata, akan sulit membuat kriptanalis melakukan analisis frekuensi, karena frekuensi karakter hasil enkripsi merata dan tidak mirip dengan plain teks.

Berdasarkan analisis yang dilakukan dapat disimpulkan bahwa SBS kuat terhadap serangan analisis frekuensi karena distribusi frekuensi karakter plain teks dan cipher teksnya sangat berbeda.

B. Serangan *Bruteforce*

Salah satu serangan yang sering digunakan oleh kriptanalis untuk memecahkan cipher teks yaitu dengan mencoba semua jenis kemungkinan kunci untuk melakukan dekripsi terhadap cipher teks. Serangan ini juga disebut dengan serangan *bruteforce* atau *exhaustive search*. Dengan menggunakan serangan ini, kunci dapat ditemukan oleh penyerang. Namun, metode ini akan menghabiskan waktu yang sangat lama karena jumlah kemungkinan kunci sangatlah banyak.

Algoritma SBS menggunakan panjang kunci panjang minimal 64 bit. Hal tersebut mengakibatkan minimal 2^{64} atau $1.8446744073709552 \times 10^{19}$ kemungkinan kunci yang harus dicoba oleh penyerang. Mesin yang kami gunakan untuk melakukan pengujian mampu menghabiskan 0.1 detik untuk melakukan dekripsi. Dengan menggunakan mesin yang kami gunakan untuk pengujian, serangan *bruteforce* akan menghabiskan waktu selama $10^{-1} * 1.8446744073709552 \times 10^{19} = 1.8446744073709552 \times 10^{18}$ detik atau 213,503,982,334,601 hari atau 584,942,417,355.072 tahun. Dengan asumsi terdapat mesin yang mampu melakukan dekripsi 100.000x lebih cepat daripada mesin pengujian yang digunakan. Waktu yang dihabiskan untuk melakukan serangan bruteforce masih $584,942,417,355.072 / (10^5 \times 10^{-1}) = 58,494,241.7355072$ tahun. Itu merupakan waktu yang sangat lama bagi seorang manusia yang masa hidupnya rata-rata 79 tahun. Perlu setidaknya 740434 generasi secara turun-temurun melakukan dekripsi agar serangan *bruteforce* berhasil mendapatkan kunci yang tepat.

Waktu yang dihabiskan untuk melakukan serangan ini sangatlah lama sehingga dapat dikatakan hampir tidak mungkin melakukan serangan *bruteforce* pada algoritma SBS.

VI. KESIMPULAN

Berdasarkan implementasi, hasil pengujian dan analisis yang dilakukan dalam disimpulkan beberapa hal sebagai berikut.

1. SBS merupakan *block cipher* dengan ukuran blok, ukuran kunci serta jumlah yang ditentukan oleh kunci yang dimasukkan pengguna dan menggunakan jaringan feistel untuk melakukan operasi kriptografinya (enkripsi dan dekripsi).
2. SBS merupakan *block cipher* yang cukup sederhana karena tidak menggunakan operasi matematika yang rumit.
3. SBS cukup kuat terhadap serangan kriptanalis analisis frekuensi
4. SBS cukup kuat terhadap serangan kriptanalis *bruteforce*
5. SBS merupakan algoritma yang cukup baik tingkat keamanannya. Hal tersebut ditunjukkan dari analisis keamanan pada bab V dimana prinsip *confusion* dan *diffusion* telah terimplementasi sehingga perubahan kunci atau *plaintext* berpengaruh besar terhadap perubahan *ciphertext* serta memiliki hubungan kunci dan *ciphertext* yang rumit.

UCAPAN TERIMA KASIH

Penulis ingin mengucapkan terima kasih kepada Tuhan Y.M.E., karena atas rahmat dan karunia-Nya lah makalah ini dapat selesai pada waktunya. Penulis juga ingin mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi Munir selaku dosen mata kuliah Kriptografi. Tidak lupa penulis juga ingin mengucapkan terima kasih kepada pihak-pihak yang secara langsung maupun tidak langsung membantu dalam merampungkan makalah ini.

DAFTAR PUSTAKA

[1] A. Menezes, P. van Oorschot, dan S. Vanstone. 1996. Handbook of Applied Cryptography. CRC Press

[2] Munir, Rinaldi. 2015. Slide Kuliah IF4020 Kriptografi: Algoritma Kriptografi Modern.

[3] Munir, Rinaldi. 2015. Slide Kuliah IF4020 Kriptografi: Pengantar Kriptografi.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 2 April 2018

Kristianto Karim
13514075

Chalvin
13514032

Lampiran

1. **S-Box**

		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S0	0	26	27	2D	27	26	3B	2E	38	33	40	3C	3D	35	2E	40	3D
	1	22	39	2A	39	31	40	30	2E	37	33	2E	21	36	23	3A	29
	2	23	25	40	3E	3D	30	3F	25	3D	26	3C	31	2A	31	33	39
	3	39	3F	22	3A	22	36	22	3D	21	23	2F	3B	22	32	2A	22
S1	0	23	31	29	3D	22	24	2E	32	33	3C	36	27	3C	31	39	22
	1	36	21	3A	26	2F	37	31	27	32	2D	29	20	3A	3F	30	24
	2	2F	3E	3D	2A	3A	3A	29	26	3C	32	20	28	40	3D	3F	2E
	3	34	3E	26	24	3C	22	26	2E	38	3E	29	33	23	21	2A	23
S2	0	27	40	23	2B	33	38	3C	24	2F	3F	21	33	32	40	21	34
	1	31	21	25	31	32	3A	30	39	3B	30	29	38	3A	3C	27	2C
	2	2A	31	40	38	29	27	2B	23	2A	2B	35	2C	26	26	3D	37
	3	23	36	39	20	37	24	22	32	2D	35	37	3B	30	20	3D	3F
S3	0	37	2A	34	31	25	21	33	3A	38	27	2C	36	3C	2F	29	34
	1	24	34	3D	21	40	36	35	22	2B	26	22	22	24	2E	36	3A
	2	2E	34	29	36	33	3C	35	3B	35	35	35	2C	30	3D	28	39
	3	40	20	31	34	30	38	3C	3B	3A	3B	34	37	36	30	2E	39
S4	0	24	3B	3E	2D	2C	34	27	2D	24	27	40	23	3D	20	22	29
	1	2E	39	39	23	2D	31	2D	20	27	3E	26	37	2E	2A	35	20
	2	37	20	22	33	28	23	38	2E	20	2E	26	24	35	37	25	3C
	3	2E	2D	28	24	3F	2E	38	38	36	30	29	27	2C	20	3D	28
S5	0	2B	2E	2F	40	2F	3A	3A	3C	2E	33	3B	29	34	33	29	2E
	1	3A	3F	2B	23	21	30	24	30	2F	3E	2D	29	2C	20	29	36
	2	23	24	25	2B	36	24	2F	25	3F	23	2F	3D	2E	27	29	31
	3	32	24	35	3C	29	32	40	36	39	27	3C	30	3F	3A	29	40
S6	0	36	37	37	32	2C	33	2B	22	2E	31	23	24	40	35	3A	26
	1	3F	2A	2F	3F	38	2E	29	31	2B	34	32	31	33	39	2B	21
	2	2E	39	3D	27	2E	24	31	35	25	2D	3F	3A	29	3B	40	2A
	3	2D	39	2B	2E	3A	2E	3B	2D	3F	24	2A	2D	31	35	31	3A
S7	0	25	39	3E	39	26	2B	2E	32	3C	36	38	30	28	30	35	34
	1	3D	34	32	20	24	20	27	22	2A	40	29	3B	3B	21	39	35
	2	23	26	2D	3C	27	3F	33	3B	32	3D	32	20	25	3E	31	35
	3	33	25	40	30	29	20	31	38	3E	2E	35	3B	21	2F	26	2D