PEMBUATAN ALGORITMA ENKRIPSI DES SMS BERBASIS MOBILE

Fettiana[1], Elvina [2], Gusti Pribadi Ratu Sajati [3]

[1] Fakultas Ilmu Komputer Universitas Gunadarma
[2,3] Fakultas Teknologi Industri Universitas Gunadarma

Email : fettiana@staff.gunadarma.ac.id, elvina@staff.gunadarma.ac.id, gusti92@yahoo.com

Abstrak

Short Message Service (SMS) merupakan sebuah revolusi teknologi komunikasi yang sangat popular. Dengan menggunakan SMS seseorang dapat saling bertukar pesan dengan orang lain. Namun, dengan adanya masalah penyadapan SMS, seseorang sudah tidak lagi mempunyai hak privasi. Sehingga pada penelitian ini, penulis mengembangkan sebuah aplikasi pada telepon selular berbasis Android yang akan merubah pesan SMS menjadi kode-kode agar isi informasi dari SMS tersebut tidak diketahui orang lain. Aplikasi ini akan mengenkripsi pesan ke dalam bentuk kode dengan key yang diinputkan oleh pengirim yang kemudian pesan kode tersebut dikirm ke nomor tujuan. Dalam penerimaan SMS, aplikasi ini akan mendekripsi pesan kode tersebut ke dalam bentuk pesan asli dengan menggunakan key yang sama dengan key pengirim. Aplikasi ini dapat dimanfaatkan oleh seseorang yang menggunakan telepon selular berbasis Android untuk mengirimkan suatu pesan penting kepada orang lain tanpa takut diketahui oleh orang yang tidak berwenang. Metode yang digunakan aplikasi ini dalam mengenkripsi dan mendekripsi pesan adalah metode blok cipher algoritma DES dan implementasinya menggunakan bahasa pemrograman Java dengan platform mobile Android.

Kata kunci : Enkripsi, Deskripsi, SMS, Android, Simetris, DES.

1. PENDAHULUAN

1.1. Latar Belakang

Sebagai media komunikasi umum, suatu jaringan sangat rawan terhadap penyadapan, pencurian, dan pemalsuan informasi. Proses pengiriman data pada suatu jaringan harus menjamin keamanan dan keutuhan, sehingga data yang dikirim dapat sampai di tujuannya. Untuk itu salah satu cara untuk mengamankan data dari kejadian-kejadian tersebut, diperlukan penyandian terhadap data yang akan dikirim.

Penyandian ini sangat penting, apalagi dalam sektor-sektor strategis seperti bisnis, perbankan, atau pemerintahan sangat memerlukan teknologi penyandian informasi. Ilmu menyandi (kriptografi) sebetulnya adalah ilmu yang sudah dikenal bahkan semenjak jaman Julius Caesar (sebelum masehi). Ilmu ini tidak hanya mencakup teknik-teknik menyandikan informasi, tetapi juga teknik untuk membongkar sandi.

Salah satu media komunikasi umum yang sering dipakai yaitu SMS (*Short Message Service*). Short Message Service merupakan sebuah pelayanan yang banyak diaplikasikan pada sistem komunikasi tanpa kabel, memungkinkan dilakukannya pengiriman pesan dalam bentuk alphanumeric antara terminal pelanggan atau antara terminal pelanggan dengan sistem eksternal seperti email, paging, voice mail, dan lain-lain. Isi sms bermacam-macam mulai dari pesan teks biasa atau umum seperti pesan pemberitahuan suatu acara sampai pesan teks yang rahasia seperti pesan suatu perjanjian maupun pemberitahuan password penting. Pesan teks rahasia yang perlu adanya keamanan data yang berupa teknik enkripsi ke dekripsi atau sebaliknya sehingga pesan dapat aman dari pihak lain yang tidak di inginkan.

DES (*Data Encryption System*) merupakan algoritma enkripsi blok simetris. DES dikatakan enkripsi blok karena pemrosesan data baik enkripsi maupun dekripsi, diimplementasikan per blok (dalam hal ini 8 byte). DES dikatakan enkripsi simetris karena algoritma yang digunakan untuk enkripsi relatif atau bahkan sama persis

dengan algoritma yang digunakan dalam proses dekripsi. Proses enkripsi dapat didefinisikan secara sederhana sebagai proses penterjemahan data "asli" yang "jelas" dan "kasat mata" yang dapat dipahami maknanya. Secara langsung menjadi data lain yang terlihat "buram" atau "acak" sehingga tidak dapat dipahami secara langsung, sedemikian rupa sehingga makna informasi yang disembunyikan tidak lagi dapat diketahui secara langsung kecuali dengan mengembalikan informasi tersebut ke bentuk aslinya. Meskipun DES bisa dianggap system penyandian yang sudah tua dibanding system penyandian kunci simetri yang lebih baru namun DES masih banyak dipakai pada system keamanan jaringan. Selain itu, DES dapat dijadikan contoh kasus yang sederhana untuk mempelajari prinsip-prinsip penyandian modern dengan kunci simetri. Untuk itu penulis mengangkat tema penelitian ini **"Pembuatan Algoritma EnkripsiDes SMS Berbasis Mobile ".**

1.2. TUJUAN

Tujuan yang dicapai pada penelitian ini adalah Pembuatan Algoritma EnkripsiDes SMS Berbasis Mobile, dengan harapan agar dapat digunakan untuk mengamankan atau menyembunyikan pesan singkat asli, sehingga pengirim tidak perlu takut pesannya akan disadap dan diketahui orang lain.

1.3. Identifikasi Masalah

Identifikasi masalah pada penelitian ini adalah untuk menerima dan mengirim pesan tanpa bisa membalas pesan. Maksimal karakter pesan yang dikirim yaitu 77 karakter. Hanya yang memiliki aplikasi yang bisa melakukan enkripsi dan deskripsi pesan. Aplikasi ini belum menyediakan layanan hapus pesan atau buku telepon. Terdapat kunci rahasia yang harus dimasukan untuk bisa mengenkripsi atau mendeskripsi pesan sms. Untuk masukan kunci rahasia yang digunakan untuk mengenkripsi sms sama dengan masukan pada saat kita mendeskripsikan sms. Hal ini karena algoritma yang digunakan yaitu algoritma simetris block cipher DES (Data Encryption Standar). Maksimal karakter kunci yang dimasukan yaitu 8 karakter.

1.4. METODE PENELITIAN

Metode penelitian yang digunakan dalam penelitian ini menggunakan metode Studi Pustaka yaitu mengambil atau mengumpulkan data atau bahan dari buku-buku, jurnal, e-book tentang kritografi khususnya algoritma DES dan Android. Serta menggunakan pendekatan metode *System Development Life Cycle* (SDLC). Yang terdiri dari beberapa fase, diantaranya :

- Pertama, fase identifikasi, fase ini merupakan fase mengidentifikasi masalah yaitu tentang keamanan mengirim pesan melalui SMS.
- Kedua, fase analisis, merupakan fase melihat kembali kebutuhan, keperluan, dan penggunaan apa saja yang akan diperlukan pada sistem yang akan dibangun.
- Ketiga, Fase perancangan, merupakan fase penggambaran model fungsional dari aplikasi yang digambarkan dengan struktur navigasi, story board dan flowchart.
- Keempat, fase implementasi, merupakan fase menerapkan hasil rancangan yang telah dibuat dengan membuat program bahasa pemrograman JAVA ANDROID dengan media Eclipse sebagai Editor.
- Kelima, fase uji coba, merupakan fase melakukan uji coba dan evaluasi terhadap aplikasi yang telah dibuat.

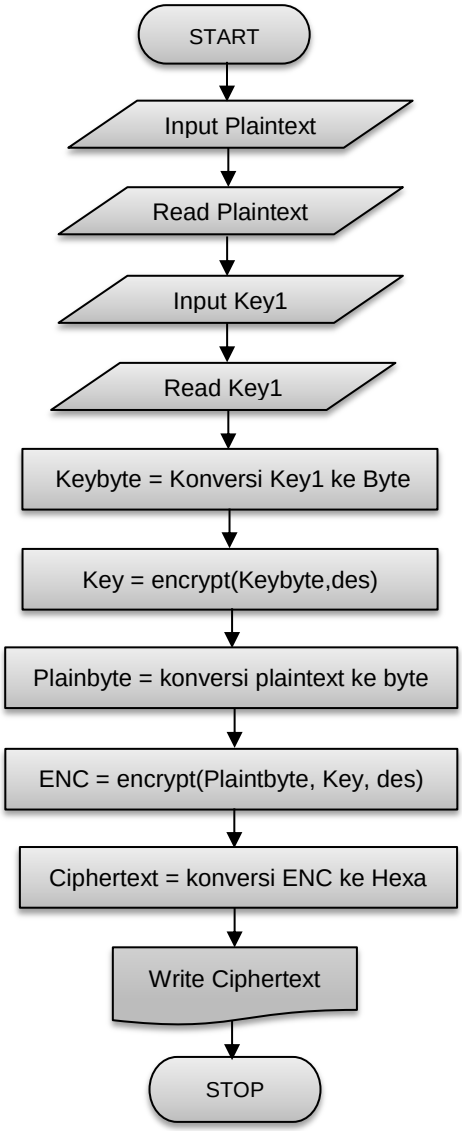
2. Short Message Service (SMS)

SMS merupakan suatu fasilitas untuk mengirim dan menerima suatu pesan singkat berupa teks melalui perangkat nirkabel, yaitu perangkat komunikasi telepon selular. Salah satu kelebihan dari SMS adalah biaya yang murah. Selain itu SMS menggunakan metode *store* dan *forward* sehingga keuntungan yang didapat adalah pada saat telepon selular penerima tidak dapat dijangkau atau tidak aktif, penerima tetap dapat menerima SMS ketika telepon selular tersebut sudah aktif kembali. SMS menyediakan mekanisme untuk mengirimkan pesan singkat dari dan menuju media-media *wireless* dengan menggunakan sebuah *Short Messaging Service Center* (SMSC), yang bertindak sebagai sistem yang berfungsi menyimpan dan mengirimkan kembali pesan-pesan singkat. Jaringan *wireless* menyediakan mekanisme untuk menemukan *station* yang dituju dan mengirimkan pesan singkat antara SMSC dengan *wireless station*. SMS mendukung banyak mekanisme *input* sehingga memungkinkan adanya interkoneksi dengan berbagai sumber dan tujuan pengiriman pesan yang berbeda.

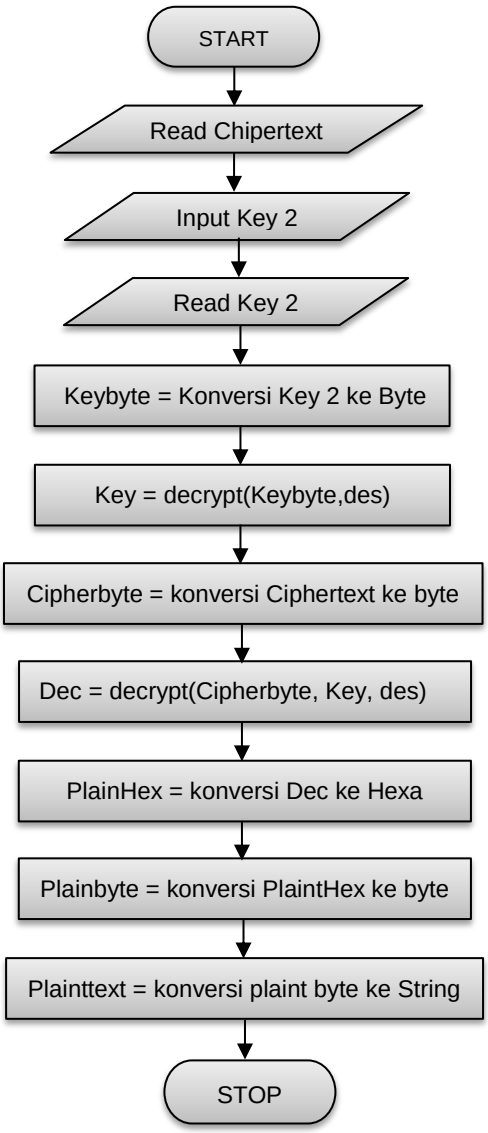
3. PEMBAHASAN

3.1. Algoritma

Tahap ini menjelaskan tentang penyajian algoritma program aplikasi EnkripsiDes SMS dengan menggunakan *flowchart.* Berikut *flowchart* dari enkripsi dan dekripsi pada program aplikasi EnkripsiDes SMS.



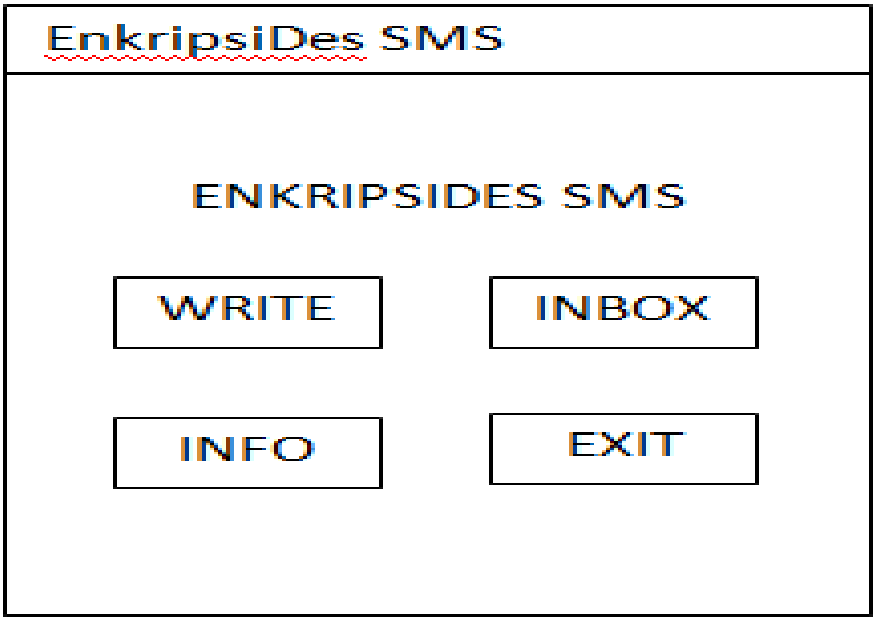
Gambar 1. Flowchart Proses Enkripsi



Gambar 2. Flowchart Proses Deskripsi

3.2. Desain *Input* dan *Output* Tampilan utama

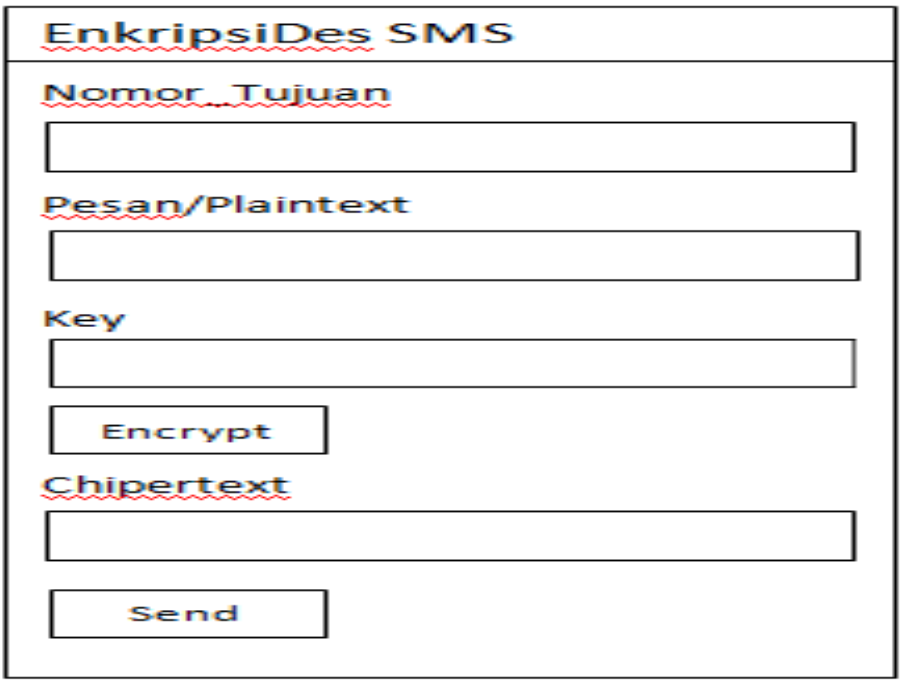
Tampilan ini merupakan tampilan awal program yang berisi *button* write message, inbox, info dan exit. Pengguna dapat memilih menu yang disediakan dari keempat *button* tersebut. Berikut rancangan tampilan utama EnkripsiDes SMS.



Gambar 3. Rancangan tampilan utama aplikasi EnkripsiDes SMS.

3.4. Tampilan Tulis Pesan

Tampilan ini digunakan untuk mengirimkan pesan, yang terdiri dari empat buah *textview*, empat buah *edittext* dan dua buah *button* yaitu encrypt dan send. Pengguna dapat mengisi Nomor_tujuan, pesan, dan key. Kemudian dilakukan proses enkripsi untuk menghasilkan pesan *ciphertext*. Pesan ini yang akan dikirim ke Nomor_tujuan.



Gambar 4. Rancangan tampilan Send EnkripsiDes SMS

3.5. Tampilan Kotak Masuk

Tampilan ini digunakan untuk menerima dan mendekripsi pesan serta membaca pesan asli. Tampilan ini terdiri dari satu buah *button*, tiga buah *textview*, satu buah *listview* dan dua buah *edittext*. Pengguna dapat melihat daftar sms masuk dengan menekan *button* "Tampil Daftar SMS". Kemudian pengguna dapat mengisi key yang sama dengan key pengirim. Selanjutnya dilakukan proses dekripsi dengan menekan pesan pada daftar sms. Hasil dekripsi tersebut akan tampil di *edittext* pesan atau *plaintext*. Berikut rancangan tampilan Receive EnkripsiDes SMS.

Gambar 4. Rancangan tampilan kotak masuk EnkripsiDes SMS

3.6. Implementasi

Pembuatan aplikasi tidak terlepas dari pengunaan perangkat keras dan perangkat lunak serta perangkat untuk pengaplikasiannya seperti telepon selular (*Handphone*). Berikut spesifikasi yang digunakan :

1. **Perangkat Keras (*Hardware*)** yang digunakan :
   - Laptop Asus dengan Sistem Operasi *Windows* XP 64-Bit
   - *Processor Intel Core i3* CPU 2.2 GHz
   - RAM 4 GB.
   - *Hardisk drive* 500 GB.
2. **Perangkat Lunak (*Software*)** yang digunakan :
   - Sistem Operasi *Windows XP SP2.*
   - *Microsoft Word 2007.*
   - *Java SE Development Kit* versi 7 *Windows.*
   - *IDE Eclipse Indigo.*
   - Android *Software Development Kit (SDK).*
   - Android *Development Tools (ADT).*

Contoh Enkripsi dengan Algoritma DES

Langkah-langkah mengenkripsi data yaitu :

- Plaintext(x) = COMPUTER
- Key(k) = 13 34 57 79 9B BC DF F1

Langkah Pertama :

Ubahlah plaintext kedalam bentuk biner

C : 01000011
O : 01001111
M : 01001101
P : 01010000
U : 01010101
T : 01010100
E : 01000101
R : 01010010

Ubahlah key kedalam bentuk biner

13 : 00010011
34 : 00110100
57 : 01010111
79 : 01111001
9B : 10011011
BC : 10111100
D : 11011111
F1 : 11110001

Langkah Kedua :

Lakukan Initial Permutation (IP) pada bit plaintext menggunakan tabel IP :

Tabel 1. Initial Permutation(IP)

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**

Urutan bit plaintext urutan ke 58 ditaruh diposisi 1,
Urutan bit plaintext urutan ke 50 ditaruh di posisi 2,
Urutan bit plaintext urutan ke 42 ditaruh di posisi 3, dst. Sehingga hasil outputnya adalah :
IP(x) : 11111111 10111000 01110110 01010111 00000000 00000000 00000110 10000011
Pecah bit pada IP(x) menjadi 2 bagian yaitu:
L_0 : 11111111 10111000 01110110 01010111
(tabel IP dengan warna kuning)
R_0 : 00000000 00000000 00000110 10000011
(tabel IP dengan warna hijau)

Langkah Ketiga :

Generate kunci yang akan digunakan untuk mengenkripsi plaintext dengan menggunakan tabel permutasi kompresi PC-1, pada langkah ini terjadi kompresi dengan membuang 1 bit masing-masing blok kunci dari 64 bit menjadi 56 bit.

Tabel 2. 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**	**45**	**38**	**30**	**22**
14	**6**	**61**	**53**	**45**	**37**	**29**
21	**13**	**5**	**28**	**20**	**12**	**4**

Dapat kita lihat pada tabel diatas, tidak terdapat urutan bit 8,16,24,32,40,48,56,64 karena telah dikompress. Berikut hasil outpunya :
CD(k) : 1111000 0110011 0010101 0101111 0101010 1011001 1001111 0001111
Pecah CD(k) menjadi dua bagian kiri dan kanan, sehingga menjadi :
C_0 : 1111000 0110011 0010101 0101111
(tabel PC-1 warna kuning)
D_0 : 0101010 1011001 1001111 0001111
(tabel PC-1 warna hijau)

Langkah Keempat :
Lakukan pergeseran kiri (Left Shift) pada C_0 dan D_0, sebanyak 1 atau 2 kali berdasarkan kali putaran yang ada pada tabel putaran sebagai berikut:

Tabel 3. Left Shift

Putaran ke - i	Jumlah Pergeseran(Left Shift)
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

Untuk putaran ke 1, dilakukan pegeseran 1 bit ke kiri
Untuk putaran ke 2, dilakukan pergeseran 1 bit kekiri
Untuk putaran ke 3, dilakukan pergeseran 2 bit kekiri, dan seterusnya. Berikut hasil outputnya:
C_0 : 1111000 0110011 0010101 0101111
D_0 : 0101010 1011001 1001111 0001111
Digeser 1 bit ke kiri
C_1 : 1110000 1100110 0101010 1011111
D_1 : 1010101 0110011 0011110 0011110
Digeser 2 bit ke kiri
C_2 : 1100001 1001100 1010101 0111111

D_2 : 0101010 1100110 0111100 0111101
Digeser 2 bit ke kiri
C_3 : 0000110 0110010 1010101 1111111
D_3 : 0101011 0011001 1110001 1110101
Digeser 2 bit ke kiri
C_4 : 0011001 1001010 1010111 1111100
D_4 : 0101100 1100111 1000111 1010101
Digeser 2 bit ke kiri
C_5 : 1100110 0101010 1011111 1110000
D_5 : 0110011 0011110 0011110 1010101
Digeser 2 bit ke kiri
C_6 : 0011001 0101010 1111111 1000011
D_6 : 1001100 1111000 1111010 101010
Digeser 2 bit ke kiri
C_7 : 1100101 0101011 1111110 0001100
D_7 : 0110011 1100011 1101010 1010110
Digeser 2 bit ke kiri
C_8 : 0010101 0101111 1111000 0110011
D_8 : 1001111 0001111 0101010 1011001
Digeser 1 bit ke kiri
C_9 : 0101010 1011111 1110000 1100110
D_9 : 0011110 0011110 1010101 0110011
Digeser 2 bit ke kiri
C_{10} : 0101010 1111111 1000011 0011001
D_{10} : 1111000 1111010 1010101 1001100
Digeser 2 bit ke kiri
C_{11} : 0101011 1111110 0001100 1100101
D_{11} : 1100011 1101010 1010110 0110011
Digeser 2 bit ke kiri
C_{12} : 0101111 1111000 0110011 0010101
D_{12} : 0001111 0101010 1011001 1001111
Digeser 2 bit ke kiri
C_{13} : 0111111 1100001 1001100 1010101
D_{13} : 0111101 0101010 1100110 0111100
Digeser 2 bit ke kiri
C_{14} : 1111111 0000110 0110010 1010101
D_{14} : 1110101 0101011 0011001 1110001
Digeser 2 bit ke kiri
C_{15} : 1111100 0011001 1001010 1010111
D_{15} : 1010101 0101100 1100111 1000111
Digeser 1 bit ke kiri
C_{16} : 1111000 0110011 0010101 0101111
D_{16} : 0101010 1011001 1001111 0001111

Setiap hasil putaran digabungkan kembali menjadi C_iD_i dan diinput kedalam tabel Permutation Compression 2 (PC-2) dan terjadi kompresi data C_iD_i 56 bit menjadi CiDi 48 bit.

Tabel 5. 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**

Berikut hasil outputnya:
C_1D_1 = 1110000 1100110 0101010 1011111 1010101 0110011
0011110 0011110
K_1 = 000110 110000 001011 101111 111111 000111 000001 110010
C_2D_2 = 1100001 1001100 1010101 0111111 0101010 1100110 0111100 0111101
K_2 = 011110 011010 111011 011001 110110 111100 100111 100101
C_3D_3 = 0000110 0110010 1010101 1111111 0101011 0011001 1110001 1110101
K_3 = 010101 011111 110010 001010 010000 101100 111110 011001
C_4D_4 = 0011001 1001010 1010111 1111100 0101100 1100111
1000111 1010101
K_4 = 011100 101010 110111 010110 110110 110011 010100 011101
C_5D_5 = 1100110 0101010 1011111 1110000 0110011 0011110
0011110 1010101
K_5 = 011111 001110 110000 000111 111010 110101 001110 101000
C_6D_6 = 0011001 0101010 1111111 1000011 1001100 1111000
1111010 1010101
K_6 = 011000 111010 010100 111110 010100 000111 101100 101111
C_7D_7 = 1100101 0101011 1111110 0001100 0110011 1100011 1101010 1010110
K_7 = 111011 001000 010010 110111 111101 100001 100010 111100
C_8D_8 = 0010101 0101111 1111000 0110011 1001111 0001111
0101010 1011001
K_8 = 111101 111000 101000 111010 110000 010011 101111 111011
C_9D_9 = 0101010 1011111 1110000 1100110 0011110 0011110
1010101 0110011
K_9 = 111000 001101 101111 101011 111011 011110 011110 000001
$C_{10}D_{10}$ = 0101010 1111111 1000011 0011001 1111000 1111010
1010101 1001100
K_{10} = 101100 011111 001101 000111 101110 100100 011001 001111
$C_{11}D_{11}$ = 0101011 1111110 0001100 1100101 1100011 1101010
1010110 0110011
K_{11} = 001000 010101 111111 010011 110111 101101 001110 000110
$C_{12}D_{12}$ = 0101111 1111000 0110011 0010101 0001111 0101010
1011001 1001111
K_{12} = 011101 010111 000111 110101 100101 000110 011111 101001
$C_{13}D_{13}$ = 0111111 1100001 1001100 1010101 0111101 0101010
1100110 0111100
K_{13} = 100101 111100 010111 010001 111110 101011 101001 000001
$C_{14}D_{14}$ = 1111111 0000110 0110010 1010101 1110101 0101011
0011001 1110001
K_{14} = 010111 110100 001110 110111 111100 101110 011100 111010
$C_{15}D_{15}$ = 1111100 0011001 1001010 1010111 1010101 0101100
1100111 1000111
K_{15} = 101111 111001 000110 001101 001111 010011 111100 001010
$C_{16}D_{16}$ = 1111000 0110011 0010101 0101111 0101010 1011001
1001111 0001111
K_{16} = 110010 110011 110110 001011 000011 100001 011111 110101

Langkah Kelima :
Pada langkah ini, kita akan meng-ekspansi data R_{i-1} 32 bit menjadi R_i 48 bit sebanyak 16 kali putaran dengan nilai perputaran 1 <= i <= 16 menggunakan Tabel Ekspansi (E).

Tabel 6. Ekspansi(E)

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**

Hasil $E(R_{i-1})$ kemudian di XOR dengan K_i dan menghasilkan Vektor Matriks A_i.

Iterasi 1
E(R(1)-1) = 100000 000000 000000 000000 000000 001101 010000 000110
K1 = 000110 110000 001011 101111 111111 000111 000001 110010
----------XOR
A1 = 100110 110000 001011 101111 111111 001010 010001 101000
Iterasi – 2
E(R(2)-1) = 011010 101110 100001 010110 100110 100101 010000 001101
K2 = 011110 011010 111011 011001 110110 111100 100111 100101
----------XOR
A2 = 000100 110100 011010 001111 010000 011001 110111 101000
Iterasi – 3
E(R(3)-1) = 010001 010111 111011 110011 110001 010101 010010 100001
K3 = 010101 011111 110010 001010 010000 101100 111110 011001
----------XOR
A3 = 000100 001000 001001 111001 100001 111001 101100 111000
Iterasi – 4
E(R(4)-1) = 010111 110001 010111 110011 110101 011100 001111 110001
K4 = 011100 101010 110111 010110 110110 110011 010100 011101
----------XOR
A4 = 001011 011011 100000 100101 000011 101111 011011 101100
Iterasi – 5
E(R(5)-1) = 110110 101001 011100 000101 011001 011010 100110 100011
K5 = 011111 001110 110000 000111 111010 110101 001110 101000
----------XOR
A5 = 101001 100111 101100 000010 100011 101111 101000 001011
Iterasi – 6
E(R(6)-1) = 100101 011011 110001 010110 101110 101100 000111 111010
K6 = 011000 111010 010100 111110 010100 000111 101100 101111
----------XOR
A6 = 111101 100001 100101 101000 111010 101011 101011 010101
Iterasi – 7
E(R(7)-1) = 110010 100001 011111 110010 100111 111101 011001 010011
K7 = 111011 001000 010010 110111 111101 100001 100010 111100
----------XOR
A7 = 001001 101001 001101 000101 011010 011100 111011 101111
Iterasi – 8
E(R(8)-1) = 111100 001010 101001 010101 010011 110000 001010 100011
K8 = 111101 111000 101000 111010 110000 010011 101111 111011
----------XOR
A8 = 000001 110010 000001 101111 100011 100011 100101 011000

Iterasi – 9
E(R(9)-1)= 010010 101111 111000 000000 000010 101111 110101 010001
K9 = 111000 001101 101111 101011 111011 011110 011110 000001
----------XOR
A9 = 101010 100010 010111 101011 111001 110001 101011 010000
Iterasi – 10
E(R(10)-1)= 100111 111000 001110 100010 100111 110111 111000 001010
K10 = 101100 011111 001101 000111 101110 100100 011001 001111
----------XOR
A10 = 001011 100111 000011 100101 001001 010011 100001 000101
Iterasi – 11
E(R(11)-1)= 010011 110111 111010 101010 101111 110011 110001 011001
K11 = 001000 010101 111111 010011 110111 101101 001110 000110
---------- XOR
A11 = 011011 100010 000101 111001 011000 011110 111111 011111

Iterasi – 12
E(R(12)-1)= 001001 011010 101001 011111 110001 010111 110010 101100
K12 = 011101 010111 000111 110101 100101 000110 011111 101001
---------- XOR
A12 = 010100 001101 101110 101010 010100 010001 101101 000101
Iterasi – 13
E(R(13)-1)= 100110 100111 110111 111011 111110 101110 101100 001010
K13 = 100101 111100 010111 010001 111110 101011 101001 000001
----------XOR
A13 = 000011 011011 100000 101010 000000 000101 000101 001011
Iterasi – 14
E(R(14)-1)= 111001 010111 110000 001000 001000 001000 001011 111011
K14 = 010111 110100 001110 110111 111100 101110 011100 111010
----------XOR
A14 = 101110 100011 111110 111111 110100 100110 010111 000001
Iterasi – 15
E(R(15)-1)= 000110 101100 001100 000001 011001 011010 100101 010100
K15 = 101111 111001 000110 001101 001111 010011 111100 001010
----------XOR
A15 = 101001 010101 001010 001100 010110 001001 011001 011110
Iterasi – 16
E(R(16)-1)= 101101 011101 010100 000101 010101 010001 010110 100010
K16 = 110010 110011 110110 001011 000011 100001 011111 110101
----------XOR
A16 = 011111 101110 100010 001110 010110 110000 001001 010111

Langkah Keenam :

Setiap Vektor A_i disubstitusikan kedelapan buah S-Box(Substitution Box), dimana blok pertama disubstitusikan dengan S_1, blok kedua dengan S_2 dan seterusnya dan menghasilkan output vektor B_i 32 bit.

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
01	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
10	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
11	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Cara menggunakan S-Box :
Kita ambil contoh S1, kemudian konversi setiap angka didalam tabel S1 yang berwarna putih menjadi biner, sehingga menjadi bentuk seperti dibawah:

	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1110	0100	1101	0001	0010	1111	1011	1000	0011	1010	0110	1100	0101	1001	0000	0111
01	0000	1111	0111	0100	1110	0010	1101	0001	1010	0110	1100	1011	1001	0101	0011	1000
10	0100	0001	1110	1000	1101	0110	0010	1011	1111	1100	1001	0111	0011	1010	0101	0000
11	1111	1100	1000	0010	0100	1001	0001	0111	0101	1011	0011	1110	1010	0000	0110	1101

Kemudian kita ambil sampel blok bit pertama dari A_1
yaitu **1**00110**0**
Kita pisahkan blok menjadi 2 yaitu:

- Bit pertama dan terakhir yaitu 1 dan 0 digabungkan menjadi 10
- Bit kedua hingga ke lima 0011

Kemudian dibandingkan dengan memeriksa perpotongan antara keduanya didapatkan nilai 1000(warna merah) dan seterusnya untuk blok kedua hingga blok kedelapan kita bandingkan dengan S2 hingga S8. Berdasarkan cara diatas diperoleh hasil sebagai berikut:

B_1 =	1000	0101	0100	1000	0011	0010	1110	1010
B_2 =	1101	1100	0100	0011	1000	0000	1111	1001
B_3 =	1101	0110	0011	1100	1011	0110	0111	1111
B_4 =	0010	1001	1101	0000	1011	1010	1111	1110
B_5 =	0100	0001	0011	1101	1000	1010	1100	0011
B_6 =	0110	1101	1101	1100	0011	0101	0100	0110
B_7 =	1110	0011	0110	1011	0000	0101	0010	1101
B_8 =	0000	1000	1101	1000	1000	0011	1101	0101
B_9 =	0110	1110	1110	0001	1010	1011	0100	1010
B_{10} =	0010	0001	0111	0000	0100	0001	0110	1101
B_{11} =	0101	1110	0000	1100	1101	1011	1100	0010
B_{12}=	0110	1000	0000	1011	0011	0110	1010	1101
B_{13} =	1111	1001	1101	1011	0010	0100	1011	0011
B_{14} =	1011	1000	0111	1110	1100	0101	1100	0001
B_{15}=	0100	0001	0011	1001	1111	0111	0010	0111
B_{16} =	1000	0001	0110	1010	1111	0111	0100	1011

Langkah Ketujuh:

Setelah didapatkan nilai vektor B_i, langkah selanjutnya adalah memutasikan bit vektor B_i menggunakan tabel P-Box, kemudian dikelompokkan menjadi 4 blok dimana tiap-tiap blok memiliki 32 bit data. Sehingga hasil yang didapat adalah sebagai berikut:

$P(B_1)$ =	00101000	10110011	01000100	11010001
$P(B_2)$ =	10001011	11011001	10001100	00010011
$P(B_3)$ =	01101111	10110010	10011100	11111110
$P(B_4)$ =	00111111	00111011	01000111	10100001
$P(B_5)$ =	10010101	00110010	11011000	01000101
$P(B_6)$ =	00100100	00011011	11110011	11111000
$P(B_7)$ =	11001000	11000001	11101110	01101100
$P(B_8)$ =	00000111	00111001	00101001	01100001
$P(B_9)$ =	11011001	00111011	10100011	10010100

$P(B_{10})$ = 00001100	00010101	01101110	00100100
$P(B_{11})$ = 01110001	00111110	10110000	01010011
$P(B_{12})$ = 10101000	01101000	10001110	11101001
$P(B_{13})$ = 10000110	11001011	11001111	11001011
$P(B_{14})$ = 00000101	11011101	00111010	01001111
$P(B_{15})$ = 10100101	00100110	11101100	11101100
$P(B_{16})$ = 00101001	11110111	01101000	11001100

Hasil P(Bi) kemudian di XOR kan dengan Li-1 untuk mendapatkan nilai Ri. Sedangkan nilai Li sendiri diperoleh dari Nilai Ri-1 untuk nilai 1 <= i <= 16.

L0 = 11111111 10111000 01110110 01010111
R0 = 00000000 00000000 00000110 10000011

P(B1) = 00101000 10110011 01000100 11010001
L(1)-1 = 11111111 10111000 01110110 01010111
----------XOR
R1 = 11010111 00001011 00110010 10000110

P(B2) = 10001011 11011001 10001100 00010011
L(2)-1 = 00000000 00000000 00000110 10000011
----------XOR
R2 = 10001011 11011001 10001010 10010000

P(B3) = 01101111 10110010 10011100 11111110
L(3)-1 = 11010111 00001011 00110010 10000110
----------XOR
R3 = 10111000 10111001 10101110 01111000

P(B4) = 00111111 00111011 01000111 10100001
L(4)-1 = 10001011 11011001 10001010 10010000
----------XOR
R4 = 10110100 11100010 11001101 00110001

P(B5) = 10010101 00110010 11011000 01000101
L(5)-1 = 10111000 10111001 10101110 01111000
----------XOR
R5 = 00101101 10001011 01110110 00111101

P(B6) = 00100100 00011011 11110011 11111000
L(6)-1 = 10110100 11100010 11001101 00110001
----------XOR
R6 = 10010000 11111001 00111110 11001001

P(B7) = 11001000 11000001 11101110 01101100
L(7)-1 = 00101101 10001011 01110110 00111101
----------XOR
R7 = 11100101 01001010 10011000 01010001

P(B8) = 00000111 00111001 00101001 01100001
L(8)-1= 10010000 11111001 00111110 11001001
----------XOR
R8 = 10010111 11000000 00010111 10101000

P(B9) = 11011001 00111011 10100011 10010100
L(9)-1 = 11100101 01001010 10011000 01010001
----------XOR
R9 = 00111100 01110001 00111011 11000101

P(B10) = 00001100 00010101 01101110 00100100
L(10)-1 = 10010111 11000000 00010111 10101000
----------XOR
R10 = 10011011 11010101 01111001 10001100

P(B11) = 01110001 00111110 10110000 01010011
L(11)-1 = 00111100 01110001 00111011 11000101
----------XOR
R11 = 01001101 01001111 10001011 10010110

P(B12) = 10101000 01101000 10001110 11101001
L(12)-1 = 10011011 11010101 01111001 10001100
----------XOR
R12 = 00110011 10111101 11110111 01100101

P(B13) = 10000110 11001011 11001111 11001011
L(13)-1 = 01001101 01001111 10001011 10010110
----------XOR
R13 = 11001011 10000100 01000100 01011101

P(B14) = 00000101 11011101 00111010 01001111
L(14)-1 = 00110011 10111101 11110111 01100101
----------XOR
R14 = 00110110 01100000 11001101 00101010

P(B15) = 10100101 00100110 11101100 11101100
L(15)-1 = 11001011 10000100 01000100 01011101
----------XOR
R15 = 01101110 10100010 10101000 10110001

P(B16) = 00101001 11110111 01101000 11001100
L(16)-1 = 00110110 01100000 11001101 00101010
----------XOR
R16 = 00011111 10010111 10100101 11100110

L16 = 01101110 10100010 10101000 10110001

Langkah Kedelapan:

Langkah terakhir adalah menggabungkan R_{16} dengan L_{16} kemudian dipermutasikan untuk terakhir kali dengan tabel Invers Initial Permutasi(IP^{-1}).

Tabel 3.8 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

Sehingga Input :
$R_{16}L_{16}$ = 00011111 10010111 10100101 11100110 01101110 10100010 10101000 10110001
Menghasilkan Output:
Cipher(dalam biner) = **01010110 11110001 11010101 11001000 01010010 10101111 10000001 00111111**
atau
Cipher(dalam hexa) = **56 f1 d5 c8 52 af 81 3f**

4. PENUTUP

Berdasarkan uraian dan pembahasan yang telah dikemukakan mengenai Pembuatan Aplikasi EnkripsiDes SMS ini dapat disimpulkan bahwa, aplikasi ini digunakan untuk mengamankan atau menyembunyikan pesan asli dari pihak ketiga yang ingin mengetahui isi pesan pengirim. Terdapat empat menu yaitu tulis pesan (*write*) yang digunakan untuk menulis pesan text asli (*plaintext*) dan mengubahnya menjadi pesan dalam bentuk tersembunyi (*ciphertext*) dengan memasukan kunci rahasia sebelum menenkripsi pesan singkat tersebut, kotak masuk (*inbox*) digunakan untuk melihat daftar pesan singkat yang masuk serta mengubahnya kembali menjadi pesan text asli menggunakan kata kunci yang sama pada saat pesan singkat dikirim sehingga pesan dapat dibaca dan dimengerti, info berisi informasi tentang penggunaan aplikasi EnkripsiDes SMS dan keluar (*exit*) yaitu untuk keluar dari program aplikasi.

DAFTAR PUSTAKA

[1] Agus Sumin, Suryadi H.S, *Pengantar Algoritma dan Pemrograman*, Gunadarma, 1997.

[2] Ali Zaki, Edy Winarno, SmitDev Community, *Membuat Sendiri Aplikasi Android untuk Pemula*, PT. Elex Media Komputindo, 2011.

[3] Ali Zaki, Edy Winarno, SmitDev Community, Hacking & *Programming dengan Android SDK untuk Advanced*, PT. Elex Media Komputindo, 2011.

[4] Yusuf Kurniawan, MT, *Kriptografi Keamanan Internet dan Jaringan Komunikasi*, Informatika Bandung, Bandung, 2012.

[5] Rifki Sadikin, *Kriptografi Untuk Keamanan Jaringan*, Andi, 2012.

[6] Anonim,"DES",http://id.wikipedia.org/wiki/ Data_Encryption_Standard, diakses:Juli 2013.

[7] Anonim, "DES", http://makalah-update.blogspot.com/2012/11/makalah-pengertian-dan-sejarah-des-data.html, diakses : Juli 2013.

[8] Anonim, "SMS Aplication", http://www.c-sharpcorner.com/UploadFile/ef3808/simple-sms-application-in-android/, diakses : Agustus 2013.

[9] Anonim,"SMS Encryption", https://github.com /herupurwito/Android_SMSEncryption, diakses:Agustus 2013.

[10] Anonim, "Pengertian SMS", http://globalonlinebook.blogspot.com/2009/09/pengertian-sms.html, diakses Agustus 2013.

[11] Anonim, "Belajar Android", http://www.vikrie.net/2013/03/belajar-android-1-persiapan-sebelum.html, diakses : Agustus 2013.

[12] Anonim, "Algoritma DES", *http://ilmu-kriptografi.blogspot.com/2009/05/algoritma-des-data-encryption-standart.html* , diakses : Agustus 2013.