

IMPLEMENTASI ALGORITMA GOST (GOSUNDARSTVENNY STANDARD) UNTUK ENKRIPSI SMS PADA TELEPON SELULAR BERBASIS ANDROID

Oleh :
Agussismery Wagi I.
Taufik Rachman

Abstrak - Masalah keamanan dan kerahasiaan data merupakan hal yang sangat penting baik dalam suatu organisasi yang berupa komersial (perusahaan), perguruan tinggi, lembaga pemerintahan, maupun dalam hal individual (pribadi). Apalagi kalau data tersebut berada dalam suatu jaringan komputer yang terhubung / terkoneksi ke jaringan publik atau internet. Kemampuan untuk mengakses dan menyediakan informasi secara cepat dan akurat akan sangat berpengaruh terhadap suatu organisasi tersebut. Kita sebisa mungkin menjaga agar data tersebut jangan sampai di baca apalagi di bajak oleh pihak yang tidak berwenang. Apabila hal sampai terjadi maka hal yang paling buruk adalah memungkinkan data yang kita punyai akan rusak (diganti) atau juga bahkan bisa hilang, dan akan sangat menimbulkan kerugian materi yang besar bagi kita

Sistem keamanan yang akan dibahas adalah sistem keamanan menggunakan algoritma GOST yang meliputi proses enkripsi, deskripsi dan contoh simulasi data dengan menggunakan algoritma GOST tersebut. Dengan enkripsi, data kita disandikan (encrypted) dengan menggunakan sebuah kunci (key). Untuk membuka (decrypt) data tersebut digunakan juga sebuah kunci yang sama dengan kunci mengenkripsi tadi atau yang sering disebut dengan private key kriptografi

Dengan melakukan enkripsi pada pesan SMS, maka keamanan pesan SMS akan lebih terjaga dan aman. Kriptografi memiliki banyak teknik dalam melakukan pengenkripsian pesan SMS dan salah satu yang memiliki tingkat keamanan yang tinggi dan tingkat kesulitan dalam pemecahannya adalah algoritma GOST. Pada saat melakukan pengenkripsian pesan harus memasukkan kunci yang akan mengamankan pesan SMS. Sehingga orang atau pihak yang tidak bertanggung jawab akan mengalami kesulitan dalam melakukan tindakannya

Kata kunci : *Gosundarstvenny Standard, Enkripsi.*

I : PENDAHULUAN

1.1 Latar Belakang

Banyak Bank di seluruh dunia menggunakan SMS untuk melakukan beberapa layanan perbankan. Salah satu cara untuk menyelesaikan permasalahan penyadapan pesan SMS adalah dengan memberikan keamanan dengan metode enkripsi. Sebelum SMS dikirimkan dilakukan enkripsi dan setelah penerima menerima pesan akan dilakukan deskripsi agar dapat membaca isi pesan SMS tersebut. Ada banyak metode enkripsi salah satunya adalah GOST (Gosudarstvenny Standard). GOST merupakan algoritma sederhana, fungsi digunakan merupakan fungsi yang sederhana dan hanya mengandalkan prinsip iterated cipher untuk keamanan.

Algoritma gost ini mempunyai kecepatan yang cukup baik dibandingkan dengan algoritma IDEA. Algoritma GOST adalah algoritma kriptografi yang dapat mengenkripsi dan mendekripsi blok data sepanjang 256 bit dan memiliki jumlah putaran yang banyak yaitu 20 kali putaran, sehingga dengan adanya keunggulan yang dimiliki oleh GOST membuat para kriptanalisis mengalami kesulitan untuk menyadap sebuah pesan.

Berdasarkan analisa yang telah dilakukan, algoritma GOST berhasil diimplementasikan pada file text, file image, dan file database. Pada penelitian di

sebuah universitas swasta Christian University di Jawa Barat oleh Anggeriawati algoritma GOST sudah pernah di implementasikan untuk menenkripsi dan mendeskripsikan sebuah dokumen. (Sinta,2009)

1.2 Rumusan Masalah

- 1 Bagaimana cara mengamankan pesan SMS dengan algoritma enkripsi GOST.
- 2 Bagaimana membangun aplikasi pada telepon selular yang mampu melakukan proses enkripsi dan deskripsi SMS menggunakan metode GOST.

1.3 Batasan Masalah

Agar pembahasan tidak menyimpang, perlu dibuat suatu batasan masalah sebagai berikut:

- 1 Software ini dibuat untuk Smartphone berbasis android.
- 2 Metode enkripsi yang digunakan adalah metode GOST.
- 3 Bahasa pemrograman yang digunakan adalah Java dengan IDE Eclipse.
- 4 Aplikasi yang dibuat hanya digunakan untuk mengirimkan dan menerima pesan teks, bukan untuk mengirimkan gambar, suara, dan lain-lain.
- 5 Panjang SMS yang bisa di enkrip per 64 bit.
- 6 SMS yang dikirim hanya untuk satu kunci dan satu nomor tujuan saja.

II : TINJAUAN PUSTAKA

2.1 Pengertian Kriptografi

Kriptografi (*Cryptography*) berasal dari Bahasa Yunani: “*Cryptos*” artinya “*Secret*” (rahasia), sedangkan “*Graphein*” artinya “*writting*” (tulisan). Jadi, kriptografi berarti “*secret writting*” (tulisan rahasia). Ada beberapa definisi kriptografi yang telah dikemukakan di dalam berbagai 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 sekadar *privacy*, tetapi juga untuk tujuan *data integrity*, *authentication*, dan *non-repudiation*. (Ariyus, 2008).

2.1.1 Pesan, Plainteks, dan Cipherteks

Pesan (*message*) adalah data atau informasi yang dapat dibaca dan dimengerti maknanya, nama lain untuk pesan adalah *plaintext* atau teks-jelas (*cleartext*). Pesan dapat berupa data atau informasi yang dikirim (melalui kurir, saluran telekomunikasi, dsb) atau yang disimpan di dalam media perekaman (kertas, *storage*, dsb). Pesan yang tersimpan tidak hanya berupa teks, tetapi juga dapat berbentuk citra (*image*), suara/bunyi (audio), dan video, atau berkas biner lainnya. (Nazruddin, 2008).

2.1.2 Pengirim dan penerima

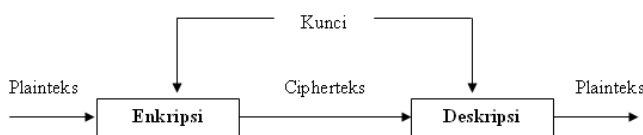
Komunikasi data melibatkan pertukaran pesan antara dua entitas. Pengirim (*sender*) adalah entitas yang mengirim pesan kepada entitas lainnya. Penerima (*receiver*) adalah entitas yang menerima pesan. Entitas di sini dapat berupa orang, mesin (komputer), kartu kredit, dan sebagainya. Jadi, orang bisa bertukar pesan dengan orang lainnya (contoh : Alice berkomunikasi dengan Bob), sedangkan di dalam jaringan komputer mesin (komputer) berkomunikasi dengan mesin (contoh: mesin ATM berkomunikasi dengan komputer *server* di bank). (Nazruddin, 2008).

2.1.3 Enkripsi dan deskripsi

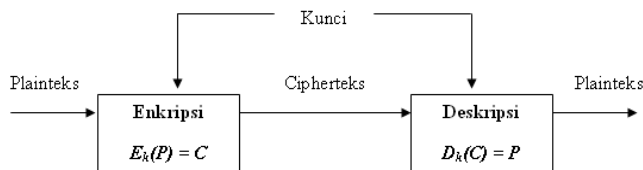
Proses menyandikan *plaintext* menjadi *ciphertext* disebut enkripsi (*encryption*) atau *enciphering* (standard nama menurut ISO 7498-2). Sedangkan proses mengembalikan *ciphertext* menjadi *plaintext* semula dinamakan deskripsi (*decryption*) atau *deciphering* (standard nama menurut ISO 7498-2). Enkripsi dan deskripsi dapat diterapkan baik pada pesan yang dikirim maupun pada pesan tersimpan. Istilah *encryption of data in motion* mengacu pada enkripsi pesan yang di transmisikan melalui saluran komunikasi, sedangkan istilah *encryption of data at-rest* mengacu pada enkripsi dokumen yang disimpan di dalam *storage*. (Nazruddin, 2008).

2.1.4 Ciper dan kunci

Ciphertext. Enkripsi dan deskripsi merupakan fungsi yang memetakan elemen-elemen antara kedua himpunan tersebut. Misalkan P menyatakan plaintext dan C menyatakan ciphertext, maka fungsi enkripsi E memetakan P ke C . (Munir, 2006) Gambar 2.1 memperlihatkan skema enkripsi dan deskripsi dengan menggunakan kunci, sedangkan gambar 2.2 mengilustrasikan enkripsi dan deskripsi terhadap sebuah pesan



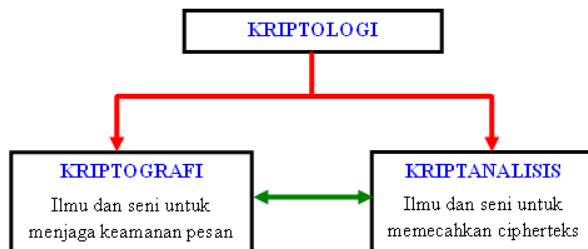
Gambar 2.1 Skema Enkripsi dan Deskripsi



Gambar 2.2 Skema rumus Enkripsi dan Deskripsi

2.1.5 Kriptanalisis dan kriptologi

Kriptografi berkembang sedemikian rupa sehingga melahirkan bidang yang berlawanan yaitu kriptanalisis. Kriptanalisis (Cryptanalysis) adalah ilmu dan seni untuk memecahkan ciphertexts menjadi plaintexts tanpa mengetahui kunci yang digunakan. Pelakunya disebut kriptanalisis. Jika seorang kriptografer (cryptographer) mentransformasikan plaintexts menjadi ciphertexts dengan suatu algoritma dan kunci maka sebaliknya seorang kriptanalisis berusaha untuk memecahkan ciphertexts tersebut untuk menemukan plaintexts atau kunci. Kriptologi (cryptology) adalah studi mengenai kriptografi dan kriptanalisis. Baik kriptografi maupun kriptanalisis keduanya saling berkaitan. (Munir, 2006)



Gambar 2.3 Pohon kriptologi

2.1.6 Proses Pembangkitan Kunci Internal

Kunci internal pada algoritma GOST dibangkitkan dari kunci eksternal yang diberikan oleh pengguna. Pembangkitan kunci internal dilakukan dengan membagi kunci eksternal 256 bit ($k_1, k_2, k_3, k_4, \dots, k_{256}$) ke dalam delapan bagian yang masing-masing panjangnya 32 bit. Pembagiannya adalah sebagai berikut:

$K_0 = (k_{32}, \dots, k_1)$

$K_1 = (k_{64}, \dots, k_{33})$

$K_2 = (k_{96}, \dots, k_{65})$

$K_3 = (k_{128}, \dots, k_{97})$

$K_4 = (k_{160}, \dots, k_{129})$

$K_5 = (k_{192}, \dots, k_{161})$

$K_6 = (k_{224}, \dots, k_{193})$

$K_7 = (k_{256}, \dots, k_{225})$. (Munir, 2006)

2.1.7 Proses Enkripsi

Proses enkripsi pada algoritma GOST untuk satu putaran (iterasi) seperti dijabarkan berikut ini:

- 64 bit plainteks dibagi menjadi 2 buah bagian 32 bit, yaitu L_i dan R_i .
Caranya: Input $a_1(0), a_2(0), \dots, a_{32}(0), b_1(0), \dots, b_{32}(0)$
 $R_0 = a_{32}(0), a_{31}(0), \dots, a_1(0)$
 $L_0 = b_{32}(0), b_{31}(0), \dots, b_1(0)$
- $(R_i + K_i) \bmod 2^{32}$. Hasil dari penjumlahan modulo 2^{32} berupa 32 bit.
Hasil dari penjumlahan modulo 2^{32} dibagi menjadi 8 bagian, dimana masing-masing bagian terdiri dari 4 bit. Setiap bagian dimasukkan ke dalam tabel *S-Box* yang berbeda, 4 bit pertama menjadi input dari *S-Box* 0, 4 bit kedua menjadi *S-Box* 1, dan seterusnya

Tabel 2.1 S-Box Algoritma GOST

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

- Hasil yang didapat dari substitusi ke S-Box kemudian digabungkan kembali menjadi 32 bit dan kemudian dilakukan RLS (*Rotate Left Shift*/ pergeseran ke kiri) sebanyak 11 bit.
- $R_{i+1} = \text{RLS XOR } L_i$
 $L_{i+1} = R_i$ sebelum dilakukan proses

Langkah nomor 2 sampai 6 dilakukan sebanyak 32 kali (putaran). Pada langkah nomor 2 penggunaan kunci dijadwalkan penggunaanya sesuai dengan putarannya.

Tabel 2.2 Penjadualan Kunci Internal Enkripsi GOST

Putaran	0	1	2	3	4	5	6	7
Kunci Internal	K0	K1	K2	K3	K4	K5	K6	K7
Putaran	8	9	10	11	12	13	14	15
Kunci Internal	K0	K1	K2	K3	K4	K5	K6	K7
Putaran	16	17	18	19	20	21	22	23
Kunci Internal	K0	K1	K2	K3	K4	K5	K6	K7
Putaran	24	25	26	27	28	29	30	31
Kunci Internal	K7	K6	K5	K4	K3	K2	K1	K0

Untuk putaran ke-31, langkah nomor 5 dan 6 sedikit berbeda. Langkah 5 dan 6 untuk putaran 31 adalah sebagai berikut,

$R_{32} = R_{31}$ sebelum dilakukan proses

$L_{32} = L_{31} \text{ XOR } R_{31}$

Sehingga, cipherteks yang dihasilkan adalah,

$L_{32} : b(32), b(31), \dots, b(1)$

$R_{32} : a(32), a(31), \dots, a(1)$

Cipherteks = $a(1), \dots, a(32), b(1), \dots, b(32)$.

(Munir, 2006).

2.1.8 Proses Dekripsi

Proses dekripsi merupakan proses kebalikan dari proses enkripsi. Penggunaan kunci pada masing-masing putaran pada proses dekripsi adalah sebagai berikut:

Tabel 2.3 Penjadualan Kunci Internal Dekripsi GOST

Putaran	0	1	2	3	4	5	6	7
Kunci Internal	K0	K1	K2	K3	K4	K5	K6	K7
Putaran	8	9	10	11	12	13	14	15
Kunci Internal	K7	K6	K5	K4	K3	K2	K1	K0
Putaran	16	17	18	19	20	21	22	23
Kunci Internal	K7	K6	K5	K4	K3	K2	K1	K0
Putaran	24	25	26	27	28	29	30	31
Kunci Internal	K7	K6	K5	K4	K3	K2	K1	K0

Algoritma yang digunakan untuk proses dekripsi sama dengan proses enkripsi dengan aturan untuk langkah 5 dan 6 pada putaran ke-31 adalah sebagai berikut, $R_{32} = R_{31}$ sebelum dilakukan proses.

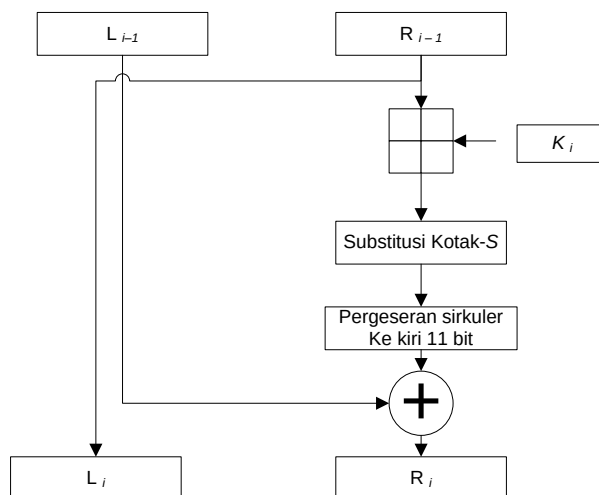
$L_{32} = R_{31} \text{ XOR } L_{31}$.

Sehingga, plainteks yang dihasilkan pada proses dekripsi adalah,

$L_{32} = b(32), b(31), \dots, b(1)$

$R_{32} = a(32), a(31), \dots, a(1)$

Plainteks = $a(1), \dots, a(32), b(1), \dots, b(32)$



Gambar 2.6 Proses Enkripsi dan Dekripsi GOST (Herryawan, 2010)

2.2 Android

Android adalah sistem operasi yang digunakan untuk telepon selular seperti telepon pintar dan komputer tablet yang berbasis Linux. Android di rilis pertama kali pada 5 Oktober 2007, hasil kerjasama Google Inc dan Android Inc, sebuah perusahaan baru yang membuat peranti lunak untuk ponsel. Karena Android mempunyai prospek yang cerah, maka untuk mengembangkan Android lebih lanjut dibentuklah Open Handset Alliance, sebuah konsorsium yang terdiri dari 34 perusahaan peranti lunak, peranti keras dan telekomunikasi termasuk didalamnya adalah Google, Motorola, Intel, Nvidia, dan masih banyak lagi. Dari hasil kerjasama tersebut akhirnya muncul 2 jenis sistem distributor Android, yaitu yang memiliki lisensi bebas (Open handset distributor) dan yang mendapat dukungan Google

Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Posel pertama yang menggunakan Android sebagai OS nya adalah HTC Dream yang dirilis pada 22 Oktober 2008. Dan hingga kini setidaknya ada sudah ada 8 jenis versi OS Android yang sudah beredar di dunia telepon selular. Jenis-jenis Android yang terbaru tersebut antara lain

a. Android versi 2.3 Gingerbread

Android Gingerbread di rilis pada 6 Desember 2010. Perubahan-perubahan umum yang didapat dari Android versi ini antara lain peningkatan kemampuan permainan (*gaming*), peningkatan fungsi *copy paste*, layar antar muka (*User Interface*) didesain ulang, dukungan format video VP8 dan WebM, efek audio baru (*reverb*, *equalization*, *headphone virtualization*, dan *bass boost*), dukungan kemampuan *Near Field Communication* (NFC), dan dukungan jumlah kamera yang lebih dari satu.

b. Android versi 3.0/3.1 Honeycomb

Android Honeycomb di rilis pada awal 2012. Merupakan versi Android yang dirancang khusus untuk *device* dengan layar besar seperti Tablet PC. Fitur baru yang ada pada Android Honeycomb antara lain yaitu dukungan terhadap *processor multicore* dan grafis dengan hardware acceleration. User Interface pada Honeycomb juga berbeda karena sudah didesain untuk tablet. Tablet pertama

yang memakai Honeycomb adalah tablet Motorola Xoom yang dirilis bulan Februari 2011. Selain itu sebuah perangkat keras produksi Asus bernama *Eee Pad Transformer* juga menggunakan OS Android honeycomb dan diharapkan akan masuk ke pasaran Indonesia pada Mei 2011

c. Android versi 4.0 ICS (Ice Cream Sandwich)

Android Ice Cream Sandwich diumumkan secara resmi pada 10 Mei 2011 di ajang Google I/O Developer Conference (San Francisco), pihak Google mengklaim Android Ice Cream Sandwich akan dapat digunakan baik di smartphone ataupun tablet. Android Ice Cream Sandwich membawa fitur Honeycomb untuk smartphone serta ada penambahan fitur baru seperti membuka kunci dengan pengenalan wajah, jaringan data pemantauan penggunaan dan kontrol, terpadu kontak jaringan sosial, perangkat tambahan fotografi, mencari email secara offline, dan berbagi informasi dengan menggunakan NFC. Ponsel pertama yang menggunakan sistem operasi ini adalah Samsung Galaxy Nexus.

d. Android versi 4.1 Jelly Bean

Android Jelly Bean juga diluncurkan pada acara Google I/O 10 Mei 2011 yang lalu. Android versi ini membawa sejumlah keunggulan dan fitur baru, diantaranya meningkatkan input keyboard, desain baru fitur pencarian, UI yang baru dan pencarian melalui Voice Search yang lebih cepat. Versi ini juga dilengkapi Google Now yang dapat memberikan informasi yang tepat pada waktu yang tepat pula. Salah satu kemampuannya adalah dapat mengetahui informasi cuaca, lalu-lintas, ataupun hasil pertandingan olahraga. Sistem operasi Android Jelly Bean 4.1 pertama kali digunakan dalam produk tablet Asus, yakni Google Nexus 7.

e. Android versi KitKat

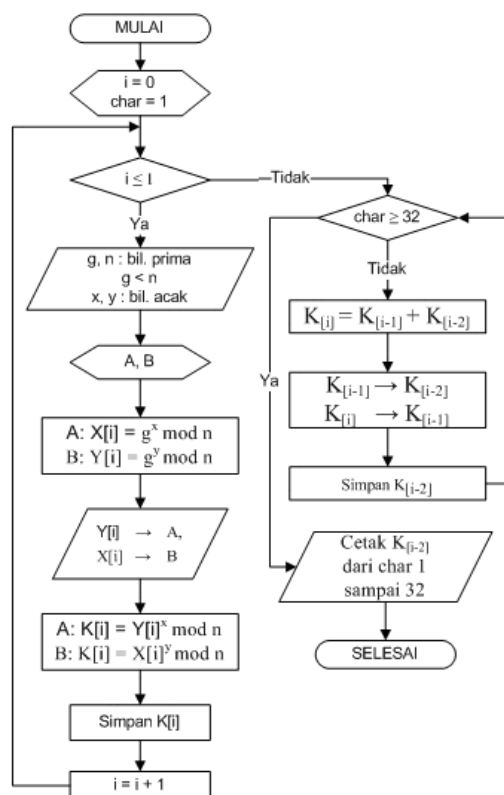
Google selaku pemilik Android telah mengumumkan peluncuran Android versi terbaru "Android KitKat" pada bulan Oktober tahun 2013, namun belum diketahui dengan pasti tanggal peluncurannya. Berbagai media banyak yang memprediksi tanggal 28 Oktober adalah tanggal peluncuran android versi 4.4 ini. Nexus 5 adalah *smartphone* pertama yang bakal mencicipi OS Android KitKat. Berikut adalah beberapa fitur Android Kitkat yang diklaim lebih cerdas dari versi android sebelumnya.

- Fitur SMS yang terintegrasi langsung kedalam Aplikasi Google Hangouts.
- Terdapat fasilitas *Cloud Printing*, dimana pengguna dapat printing secara nirkabel / mengirim perintah ke Laptop / PC yang terhubung dengan printer.
- Desain *Icon* dan tema yang lebih unik dan realistis.
- Mendengarkan perintah suara dari Google NOW tanpa menguras daya baterai.
- Navigasi dan statusbar yang mengalami pembaruan.
- Interface yang sangat halus.
- Bisa mengakses aplikasi kamera dari layar yang terkunci. (Bima, 2013)

III : PERANCANGAN SYSTEM

3.1 Flowchart Proses Pembentukan Kunci

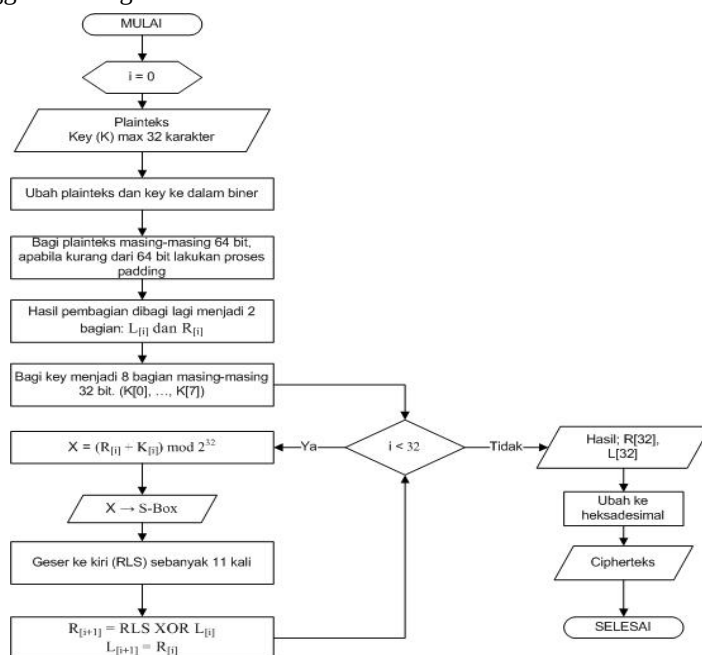
Berikut ini adalah flowchart pada proses pembentukan kunci dengan menggunakan algoritma GOST :



Gambar 3.1 Flowchart Proses Pembentukan Kunci (Yuniar, 2008)

3.2 Flowchart Proses Enkripsi Pesan

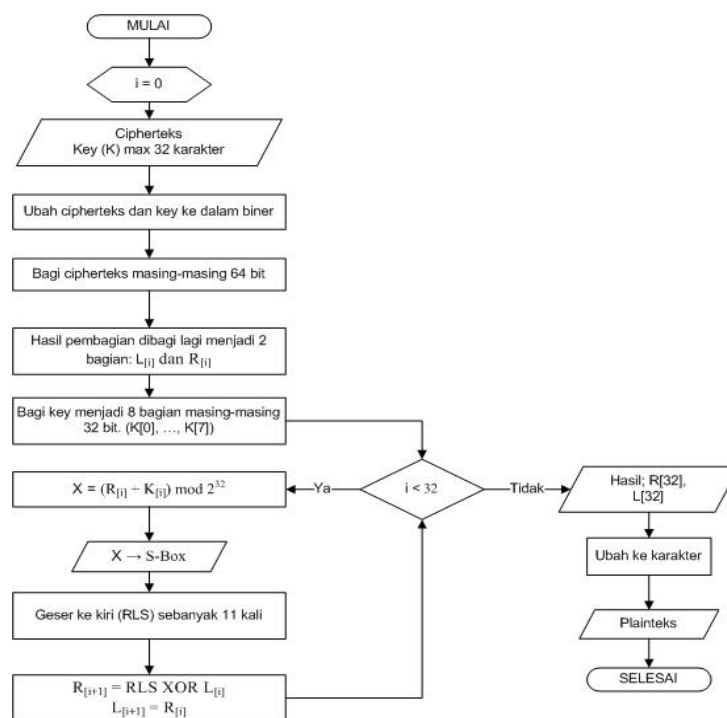
Berikut ini adalah flowchart pada proses pembentukan enkripsi pesan dengan menggunakan algoritma GOST:



Gambar 3.2 Flowchart Proses Enkripsi Pesan (Yuniar, 2008)

3.3 Flowchart Proses Dekripsi Pesan

Berikut ini adalah flowchart pada proses pembentukan dekripsi pesan dengan menggunakan algoritma GOST:



Gambar 3.5 Flowchart Proses Dekripsi Pesan

3.4 Perhitungan manual Enkripsi Algoritma GOST

Misalkan A ingin mengirimkan pesan rahasia kepada B. A dan B menyepakati kunci yang sama yaitu 7527779156 2353916261 0171643266 04. Kunci ini kemudian digunakan untuk mengenkripsi pesan A. A ingin menulis pesan “ENKRIPSI” untuk dikirimkan kepada B. Langkah-langkah yang dilakukan A adalah:

1. Pertama yang dilakukan A adalah mengubah kunci ke dalam bentuk biner berdasarkan kode ASCII, maka menjadi:
 001101111001101010011001000110111
 001101111001101110011100100110001
 00110101001101100011001000110011
 00110101001100110011100100110001
 00110110001100100011011000110001
 00110000001100010011011100110001
 00110110001101000011001100110010
 00110110001101100011000000110100
2. Bagi ke dalam 8 bagian dengan masing-masing bagian terdiri dari 32 bit.
 $K(0) = k(32), \dots, k(1) = 11101100010011001010110011101100 = 3.964.447.980$
 $K(1) = k(64), \dots, k(33) = 10001100100111001110110011101100 = 2.359.094.508$
 $K(2) = k(96), \dots, k(65) = 11001100010011000110110010101100 = 3.427.560.620$
 $K(3) = k(128), \dots, k(97) = 10001100100111001100110010101100 = 2.359.086.252$
 $K(4) = k(160), \dots, k(129) = 10001100011011000100110001101100 = 2.355.907.692$
 $K(5) = k(192), \dots, k(161) = 10001100111011001000110000001100 = 2.364.312.588$
 $K(6) = k(224), \dots, k(193) = 01001100110011000010110001101100 = 1.288.449.132$

- $K(7) = k(256), \dots, k(225) = 00101100000011000110110001101100 = 739.011.692$
3. Melakukan proses enkripsi pesan. Proses enkripsi putaran ke-0 :
 Plainteks = ENKRIPSI
 Ubah ke dalam bentuk biner =
 $0100010101001110010010110101001001001001010100000101001101001001$
 4. 64 bit plaintexts dibagi menjadi 2 bagian 32 bit, yaitu L_i dan R_i
 $L(0) = 10010010110010100000101010010010$
 $R(0) = 01001010110100100111001010100010$
 5. $R(0) + K(0) \bmod 2^{32}$
 $R(0) = 1.255.305.890$
 $K(0) = 3.964.447.980$

$$\begin{array}{r} \text{-----} + \\ 5219753870 \bmod 2^{32} = 924.786.574 \text{ diubah dalam biner} \\ 924.786.574 = 00110111000111110001111110001110 \end{array}$$
 Penggunaan kunci pada langkah ini dijadualkan berdasarkan putarannya, sesuai tabel 2.2.
 6. Pecah menjadi 8 kelompok dan masukkan ke dalam SBox (tabel 2.1).
 $0011 = 3 = \text{SBox}(0) = 2 = 0010$
 $0111 = 7 = \text{SBox}(1) = 10 = 1010$
 $0001 = 1 = \text{SBox}(2) = 8 = 1000$
 $1111 = 15 = \text{SBox}(3) = 3 = 0011$
 $0001 = 1 = \text{SBox}(4) = 12 = 1100$
 $1111 = 15 = \text{SBox}(5) = 14 = 1110$
 $1000 = 8 = \text{SBox}(6) = 0 = 0000$
 $1110 = 14 = \text{SBox}(7) = 8 = 1000$
 Hasilnya digabungkan menjadi: $00101010100000111100111000001000$
 7. Lakukan Rotate Left Shift sebanyak 11 kali
 $\text{RLS}(11) = 00011110011100000100000101010100$
 8. $R(1) = \text{RLS XOR } L(0)$
 $\text{RLS} = 00011110011100000100000101010100$
 $L(0) = 10010010110010100000101010010010$

$$\begin{array}{r} \text{----- XOR} \\ R(1) = 10001100101110100100101111000110 \end{array}$$
 9. $L(1) = R(0)$ sebelum proses.
 $L(1) = 01001010110100100111001010100010$
 Proses tersebut dilakukan sebanyak 32 putaran. Namun pada putaran ke-31 terdapat perbedaan pada langkah 8 dan 9:

Proses Enkripsi Putaran Ke-31

1. $L(31) = 00011110110111100110110101000110$
 $R(31) = 00001111000000010001111011101001$
2. $R(31) + K(0) \bmod 2^{32}$
 $R(31) = 251.731.689$
 $K(0) = 3.964.447.980$

$$\begin{array}{r} \text{-----} + \\ \text{Hasil} = 4.216.179.669 \bmod 2^{32} = 4.216.179.669 \\ 4.216.179.669 \text{ diubah dalam biner } 11111011010011011100101111010101 \end{array}$$
3. Pecah menjadi 8 kelompok dan masukkan ke dalam SBox.
 $1111 = 15 = \text{SBox}(0) = 3 = 0011$
 $1011 = 11 = \text{SBox}(1) = 1 = 0001$
 $0100 = 4 = \text{SBox}(2) = 10 = 1010$
 $1101 = 13 = \text{SBox}(3) = 2 = 0010$

1100 = 12 = SBox(4) = 0 = 0000

1011 = 11 = SBox(5) = 5 = 0101

1101 = 13 = SBox(6) = 8 = 1000

0101 = 5 = SBox(7) = 7 = 0111

Hasilnya digabungkan menjadi: 00110001101000100000010110000111

4. Lakukan Rotate Left Shift sebanyak 11 kali

RLS(11) = 00010000001011000011100110001101

5. R(32) = R(31) sebelum proses.

R(32) = 00001111000000010001111011101001

6. L(32) = RLS XOR L(31)

RLS = 00010000001011000011100110001101

L(31) = 00011110110111100110110101000110

----- XOR

L(32) = 00001110111100100101010011001011

7. L(32) = b(32), b(31), ..., b(1)

R(32) = a(32), a(31), ..., a(1)

8. Hasil = a(1), ... a(32), b(1), ... b(32)

Hasil dalam biner =

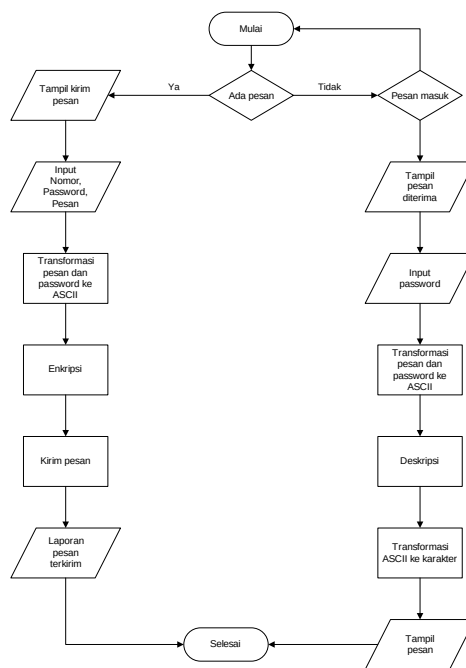
100101110111000100000001111000011010011001010100100111101110000

Ubah ke dalam heksadesimal = 977880F0D32A4F70

Kemudian A mengirimkan cipherteks tersebut ke B. (Shilwan, 2013)

3.5 Perancangan Antarmuka

Perancangan antarmuka adalah proses membuat perancangan *form-form* tampilan layar, selain itu dalam proses ini juga ditentukan bentuk dan isi dokumen sumber untuk memasukkan data yang kemudian diolah menjadi keluaran yang dapat digunakan oleh pengguna. Diagram alur aplikasi KriptoSMS dapat dilihat pada gambar 3.6



Gambar 3.4 Flowchart Aplikasi Enkripsi Deskripsi



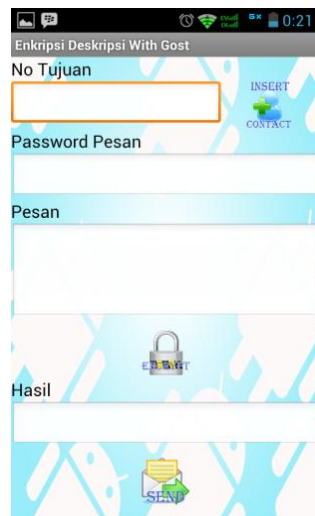
Gambar 3.5 *form* utama aplikasi

IV : IMPLEMENTASI, PENGUJIAN DAN ANALISIS HASIL

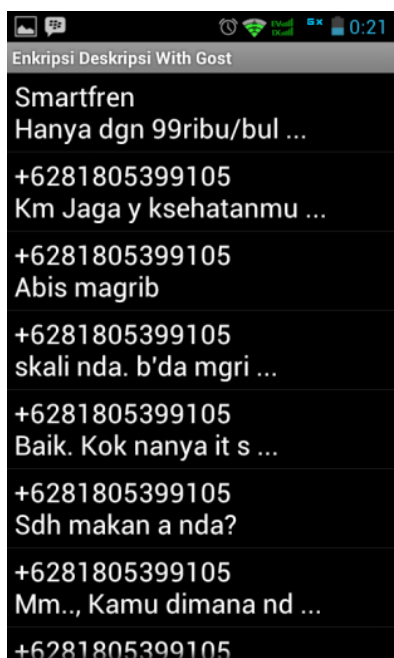
4.1 Implementasi Rancangan Antarmuka



Gambar 4.1 *form* utama aplikasi



Gambar 4.2 *form* tulis pesan



Gambar 4.3 form listview pesan masuk



Gambar 4.4 form baca pesan masuk

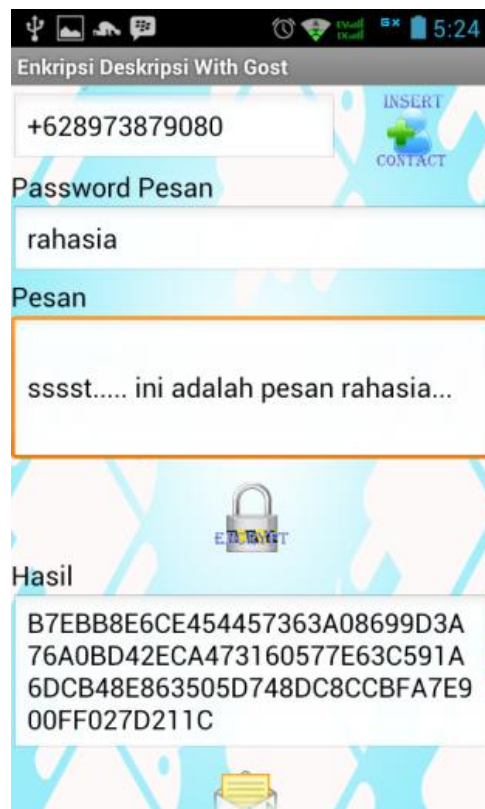
4.2 Pelaksanaan Pengujian

Pengujian aplikasi ini menggunakan metode *GOST*. Sesuai dengan material pengujian maka akan dilaksanakan pengujian sebagai berikut:

1. Pengiriman Pesan

Proses ini berfungsi untuk mengenkripsi pesan dan mengirimkan pesan ke nomor tujuan. Langkah-langkah yang dilakukan *user* dalam proses ini adalah sebagai berikut :

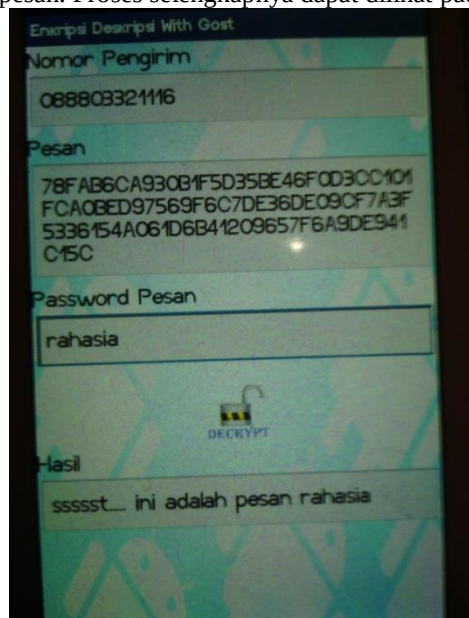
1. *Input pesan*
User diminta untuk menginputkan pesan yang akan dikirim. Pesan tersebut akan dienkripsi terlebih dahulu sebelum dikirimkan ke nomor tujuan.
2. *Input nomor tujuan*
User diminta untuk menginputkan nomor tujuan kemana pesan akan dikirimkan.
3. *Input password*
Password yang diinputkan merupakan *key* yang akan dipakai untuk mengenkripsi pesan. Proses selengkapnya dapat dilihat pada gambar 4.6.



Gambar 4.6 proses enkripsi.

2. Penerimaan Pesan

Apabila ada pesan masuk maka tampilan layar akan berubah ke layar listview. Dan *user* diminta untuk memilih pesan masuk yang ada. Pada proses ini user diminta untuk memasukkan password yang akan menjadi key untuk mendeskripsikan pesan. Proses selengkapnya dapat dilihat pada gambar 4.7.



Gambar 4.7 proses deskripsi

4.1 Analisis Hasil

Pada tahap ini akan dijelaskan analisis hasil kinerja dari aplikasi perangkat lunak KriptoSMS

a. Analisis Enkripsi Pesan

Misalkan A ingin mengirimkan pesan rahasia kepada B. A dan B menyepakati kunci yang sama yaitu “rumah”. Kunci ini kemudian digunakan untuk mengenkripsi pesan A. A ingin menulis pesan “transfer uang ke nomer babe 190456 sekarang” untuk dikirimkan kepada B.

Maka hasil dari enkripsi

C6BF7A42936BDFEA92C7A61456F092EF5E855ADDD42524316507C3BE189CC1928F3A5F0C13207AA6363F03DCD35677C9

B menerima pesan dari A berupa *cipherteks* yang bertuliskan C6BF7A42936BDFEA92C7A61456F092EF5E855ADDD42524316507C3BE189CC1928F3A5F0C13207AA6363F03DCD35677C9. Hal yang harus dilakukan B adalah mendekripsi pesan tersebut dengan memasukkan kunci yang disepakati sebelumnya. Dan hasilnya “transfer uang ke nomer babe 190456 sekarang”.

4.2 Tabel Penelitian

Plaintext :
Berbicara tentang perempuan berarti berbicara tentang ibu sosok perempuan tangguh tempat kita berteduh mengasuh peluh, dia yang paling mengerti kita saat kita jatuh,
Kunci : Mencoba Enkripsi Pesan
Panjang Karakter : 161 Karakter
Hasil Enkripsi
17Ao4DFDD5E2B5C1DiC1078D6A22B6DA5E46B136F2E239883916C49BOE6785EFo519B598A884522A57BE7E31oB169F3B331EB353D489069oF4C638D24CA287DDFBA3B278F3781169FE56D86D7916EDA2DDOEF8BAF8879AFD8BAD13511Co160B65AA4404CBF3FC47ADD9E1DF855Do567D49CB02796388AO1AB843DCC12FOEE203Fo306397CFDF6B812C794F366F814199FBFCAB4F8C6CA93C8FFOF1FE557C7349D5Co35D796B3CAOEE57A8C449A58228B

Plaintext :
Berbicara tentang perempuan berarti berbicara tentang ibu sosok perempuan tangguh tempat kita berteduh mengasuh peluh, dia yang paling mengerti kita saat kita jatuh, mengangkat dan memberikan kita semangat baru. Laki-laki dan perempuan pada dasarnya adalah satu yang ketika hilang memunculkan rasa rindu kedua dicipta bukan beradu siapa yang lebih maju tapi saling membantu dalam doa penghulu.
Kunci : Mencoba Enkripsi Pesan Dengan Penambahan Kunci
Panjang Karakter : 161 Karakter
Hasil Enkripsi
17Ao4DFDD5E2B5C1D1C1078D6A22B6DA55FDE25FAFo2A6211Aoo54B4599631FF7F497oA7AOB54EE962BB42C18D67199AFB414BC571ED215A9D48943856B154Eo649BD2B205ADBB8D2951913AA84266D9F785F8F9E2C58236ED26oCCB5DC4C2F9B67oDA506BCEA547oFC4F445BOFFE3F1C1B5828DA928B67749423433858AE6BB1552DBEFD717EOF88E3BB352B7621F2FA97666599596D567824EAA505FA6DB82943C18151FDCC4E7EA410139B62CD8304B32FD169F8DF77569E8F5E8A3EOC27D5FF6CBC2E3oF4B6A87oCBA61FCDBC17DE3EFB BEFAB93F2C29E4958E546268oEAo47D7D786CEB6A792Do645E4D5B4231572C374A948096BF4EOAC46989933CA8A7ABA61Bo177E49F2C8E59E81E2BEB41oF77D1BE93149EE1AAD93A9987FD1821C17FAED9EB3E3DCFE29D73850463F9EE35DE906B47ACEBD

V : KESIMPULAN DAN SARAN

5.1 Kesimpulan

Kesimpulan yang dapat diambil dari penulisan skripsi ini adalah sebagai berikut :

1. Aplikasi ini dapat mengirimkan dan menerima sebuah SMS sekaligus memiliki fasilitas untuk mengamankan atau menyembunyikan informasi dari pesan yang dikirim.
2. Untuk membangun aplikasi ini kita dapat menggunakan IDE Eclipse sebagai bahasa pemrograman yang dipakai dan algoritma GHOST sebagai metode yang aman dan cepat dalam proses enkripsi dan mendeskripsi.

5.2 Saran

Saran yang ingin disampaikan adalah aplikasi ini dapat dikembangkan lebih lanjut dengan cara menambahkan basis data sehingga dapat memiliki *inbox* pesan.

DAFTAR PUSTAKA

- Afrianto, Teguh. 2011. *Membuat Interface Aplikasi Android Lebih Keren Dengan LWUIT*. Andi. Yogyakarta.
- Munir, Rinaldi, 2005. *Kriptografi*, Penerbit Informatika. Bandung.
- Nazruddin.Safaat. 2007. *Aplikasi Berbasis Android*, Penerbit Informatika. Bandung
- Safaat H, Nazruddin. 2011. *Android Pemrograman Aplikasi Mobile Smartphone Dan Tablet PC Berbasis Android*. Informatika. Bandung.
- Siregar, Ivan Michael. 2011. *Membongkar Source Code Berbagai Aplikasi Android*. Gava Media. Yogyakarta
- <http://sinta.ukdw.ac.id/sinta,2009>
- <http://enam5.blogspot.com/2013/06/jenis-dan-macam-macam-versi-android.html>
- <http://rapendik.blogspot.com>
- http://shilohefrata.blogspot.com/2013/01/proses-enkripsi-plainteks-algoritma_17.html