Penerapan Algoritma Blowfish Untuk Keamanan SMS Pada Android

Tugas Akhir Mata Kuliah Keamanan Perangkat EL5215

Anton Prafanto
(23215051)

Magister Teknik Elektro
Sekolah Teknik Elektro dan Informatika
Institute Teknologi Bandung
2016

ABSTRAK

SMS (*Short Message Service*) merupakan layanan yang disediakan oleh telepon seluler untuk mengirim dan menerima pesan singkat, SMS dinilai sangat praktis, murah dan efisien. Secara umum, SMS dikirim dalam bentuk *plain text* (meskipun di encoding/deconding dengan Protocol Data Unit) tanpa terenkripsi dari pengirim ke penerima SMS. Jika terjadi penyadapan pada jalur komunikasi, maka teks SMS akan sangat mudah dibaca oleh penyadap. Dalam enskripsi data khususnya yang akan dibahas yaitu SMS, terdapat berbagai algoritma yang dapat digunakan untuk mengamankan suatu data. Salah satunya adalah algoritma Blowfish yang dapat digunakan untuk mengenkripsi sebuah data. Blowfish sendiri merupakan algoritma kunci simetris yang memiliki kunci yang sama untuk mengenkripsi dan mendekripsi sebuah data. Algoritma Blowfish termasuk kedalam chipper block dan sampai saat ini masih dianggap aman karena belum ada attack yang benar-benar mematahkan algoritma Blowfish. Selanjutnya apa yang akan dibahas pada makalah ini nantinya meliputi bagaimana algoritma Blowfish mengenkripsi sebuah SMS dan juga akan membahas bagaimana cara agar algoritma Blowfish dapat bekerja dengan optimal.

Kata kunci : SMS (Short Message Service), Blowfish, Enkripsi, Chipper Block.

1. PENDAHULUAN

Penggunaan teknologi telepon genggam (handphone) sebagai alat bantu telekomunikasi pada saat ini mengubah cara pandang masyarakat dalam berkomunikasi. Telepon genggam mempunyai beberapa fungsi komunikasi yang dapat digunakan antara lain, *video call, SMS, MMS, chatting,* internet, dan lain-lain. Berkembangnya teknologi telepon genggam dapat dilihat dengan munculnya berbagai sistem operasi yang lengkap layaknya computer, diantaranya adalah Android. Android adalah sebuah sistem operasi untuk perangkat telepon yang berbasis linux yang mencakup sistem operasi, *middleware,* aplikasi dan menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi. Android berkembang pesat karena mempunyi *platform* yang sangak lengkap baik dalam sistem operasi, aplikasi dan *tool* pengembagannya, market aplikasi serta mendapatkan dukungan yang sangat tinggi dari komunitas *open source* di dunia. Meskipun Android memiliki fitur yang lengkap, namun layanan SMS (*Short Message Service)* sebagai layanan pertukaran informasi atau pesan pendek menjadi media komunikasi favorit karena saat ini semua telepon genggam memiliki layanan ini dan yang paling penting adalah biaya SMS relatif murah. Namun demikian SMS tidak menjamin integritas dan keamanan pesan yang disampaikan. Pesan yang bersifat personal atau rahasia tidak dijamin sampai ke penerima tanpa diketahui informasinya oleh pihak yang tidak bertanggungjawab. Beberapa resiko yang dapat mengancam keamanan pesan pada layanan SMS antara lain SMS *spoofing,* SMS *snooping,* dan SMS *Interception.*

SMS bekerja dalam jaringan nirkabel. Dalam aplikasinya, pentrasmisian SMS membutuhkan beberapa komponen khusus untuk mengirimkan pesan sampai ke tujuan. Komponen yang diperlukan untuk melakukan komunikasi SMS diantaranya adalah : BTS (*Base Transceiver Station*), MSC (*Mobile Switching Center*),SMSC (*SMS Service Center*) komponen yang paling krusial adalah SMSC adalah sebuah perangkat yang terpasang pada

jaringan utama SMSC ini berfungsi untuk menerima SMS dan menelusuri nomor tujuan, dan mengirimkannya ke perangkat tujuan (Telepon Seluler). SMSC ini juga berperan sebagai penyimpanan sementara untuk SMS. Jadi, jika nomor tujuan tersebut tidak aktif, SMS tersebut akan tersimpan pada SMSC dan SMSC akan mengirimkannya kembali jika perangkat tujuan telah aktif kembali. Sebagai tambahan, SMSC akan memberikan notifikasi kepada pengirim apakah pengiriman SMS tersebut berhasil ataupun tidak. Namun, karena keterbatasan memori penyimpanan, SMSC tidak dapat menyimpan SMS untuk jangka waktu yang lama.

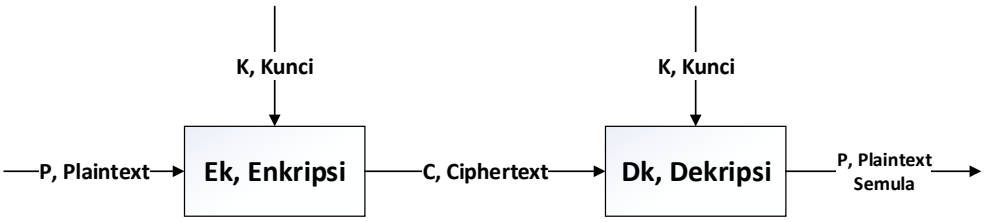
Dengan tersimpannya SMS pada SMSC, maka seorang operator dapat memperoleh informasi atau membaca SMS di dalam SMSC tersebut, hal ini dapat dibuktikan dari beberapa kasus yang ditangani kepolisian, kejaksaan atau KPK, dimana pihak penyelidik tersebut meminta transkip SMS ke operator untuk dijadikan bahan penyelidikan di persidangan. Dengan demikian dibutuhkan suatu metode dan aplikasi yang dapat mempertimbangkan solusi *encrypted end to end* dengan melakukan enkripsi terhadap pesan SMS. Enkripsi adalah proses mengubah suatu pesan asli yang disebut *plaintext* menjadi sebuah sandi atau kode yang tidak terbaca yang disebut *chipertext* dan tidak dapat dimengerti, untuk mengembalikan pesan ke bentuk asli seperti semula diperlukan proses yang disebut dekripsi. Enkripsi dimaksudkan untuk melindungi dan menyamarkan informasi agar tidak terlihat oleh pihak atau orang yang bukan seharusnya. Komunikasi yang tidak dilindungi menimbulkan kerentanan terhadap keamanan yang cukup serius karena dibeberapa kasus komunikasi menggunakan sms berisi data yang bersifat rahasia. Salah satu metode yang cukup efisien adalah penggunaan autentikasi untuk melindungi dari serangan yang tidak diinginkan selama proses transmisi pesan[1].Tujuan utama dari mekanisme keamanan adalah untuk memberikan privasi pesan yang menjamin kerahasiaan, integritas dan tidak ada pengulangan data[2].

Teknik kriptografi itu sendiri ada 2 jenis berdasarkan perbedaan teknik enkripsinya yaitu enkripsi asimetris dan enkripsi simetris. Enkripsi asimetris biasa disebut kunci enkripsi publik dimana ada 2 kunci yang digunakan untuk proses enkripsi yaitu kunci publik dan kunci privat. Sedangkan enkripsi simetris hanya menggunakan 1 kunci untuk enkripsi dan dekripsi. Pada makalah ini lebih fokus pada penggunaan algoritma kunci simetris yang dapat mengurangi masalah *overhead* komputasi dan perhitungan algoritma dan meningkatkan kinerja enkripsi. Algoritma Blowfish yang akan dibahas pada makalah ini termasuk kedalam algoritma enkripsi simetris yang berarti bahwa algoritma ini menggunakan kunci yang sama baik untuk melakukan enkripsi dan dekripsi.

2. LANDASAN TEORI

2.1 ENKRIPSI DAN DEKRIPSI

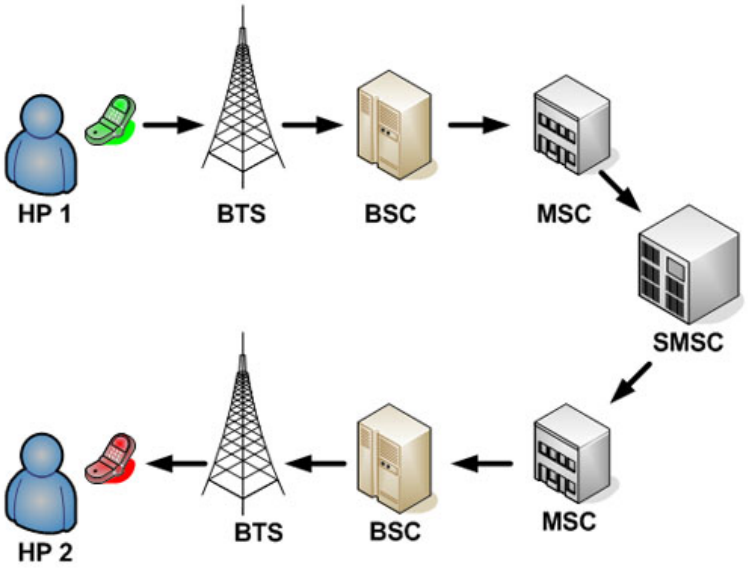
Enkripsi adalah suatu proses untuk merubah sebuah pesan, data atau informasi (biasa disebut *plaintext*), sehingga informasi tersebut tidak dapat dibaca oleh orang yang tidak bertanggung jawab (*ciphertext*) (standar nama menurut ISO 7498-2)[4]. Jadi *plaintext* adakah informasi yang dapat dimengerti dan *ciphertext* adalah informasi yang tidak dapat dimengerti atau dibaca. Sedangkan dekripsi adalah suatu proses untuk mengembalikan informasi yang sudah di enkripsi menjadi bisa dibaca kembali (standar nama menurut ISO 7498-2). Secara umum dapat digambarkan sebagai berikut :



Gambar. 1 Skema Enkripsi dan Dekripsi[5][6][7]

2.2 SMS (*SHORT MESSAGING SERVICE)*

Short Messaging Service (SMS) merupakan salah satu fitur dari GSM yang dikembangkan dan distandarisasi oleh ETSI. Pada saat kita mengirim pesan SMS dari handphone, maka pesan SMS tersebut tidak langsung dikirim ke handphone tujuan, akan tetapi terlebih dahulu dikirim ke SMS Center (SMSC) dengan prinsip *Store and Forward*, setelah itu baru dikirimkan ke handphone yang dituju. Proses pengiriman SMS dapat dilihat pada gambar 2.



Gambar 2. Skema Proses SMS

SMS bekerja dalam jaringan nirkabel. Dalam aplikasinya, pentrasmisian SMS membutuhkan beberapa komponen khusus untuk mengirimkan pesan sampai ke tujuan. Komponen yang diperlukan untuk melakukan komunikasi SMS diantaranya adalah :

a. BTS (*Base Transceiver Station*) BTS ini merupakan sebuah perangkat yang memfasilitasi komunikasi nirkabel antara perangkat user dengan jaringan. Perangkat user ini dapat meliputi telepon selular, komputer dengan koneksi internet nirkabel, dan lain-lain.
b. MSC (*Mobile Switching Center*) MSC ini adalah sebuah noda layanan pengiriman utama bagi GSM/CDMA. Perangkat ini berfungsi untuk *routing* panggilan suara, SMS, FAX, maupun *conference call*.
c. c. SMSC (*SMS Service Center*) SMSC adalah sebuah perangkat yang terpasang pada jaringan utama SMSC ini berfungsi untuk menerima SMS dan menelusuri nomor tujuan, dan mengirimkannya ke perangkat tujuan (Telepon Seluler)[3].

2.3 ALGORITMA BLOWFISH

Blowfish diciptakan oleh seorang Cryptanalyst bernama Bruce Schneier, Presiden perusahaan Counterpane Internet Security, Inc (Perusahaan konsultan tentang kriptografi dan keamanan komputer) dan dipublikasikan tahun 1994. Dibuat untuk digunakan pada komputer yang mempunyai microposesor besar (32-bit keatas dengan cache data yang besar)[8]. Blowfish merupakan algoritma yang tidak dipatenkan dan licensefree, dan tersedia secara gratis untuk berbagai macam kegunaan.

Pada saat Blowfish dirancang, diharapkan mempunyai kriteria perancangan sebagai berikut :

1. Cepat, Blowfish melakukan enkripsi data pada microprocessors 32-bit dengan rate 26 clock cycles per byte.

2. Compact (ringan), Blowfish dapat dijalankan pada memori kurang dari 5K.
3. Sederhana, Blowfish hanya menggunakan operasi-operasi sederhana: penambahan, XOR, dan lookup tabel pada operan 32-bit.
4. Memiliki tingkat keamanan yang bervariasi, panjang kunci yang digunakan oleh Blowfish dapat bervariasi dan bisa sampai sepanjang 448 bit.

Dalam penerapannya sering kali algortima ini menjadi tidak optimal. Karena strategi implementasi yang tidak tepat. Algoritma Blowfish akan lebih optimal jika digunakan untuk aplikasi yang tidak sering berganti kunci, seperti jaringan komunikasi atau enkripsi file otomatis. Selain itu, karena algoritma ini membutuhkan memori yang besar, maka algoritma ini tidak dapat diterapkan untuk aplikasi yang memiliki memori kecil seperti smartcard. Panjang kunci yang digunakan, juga mempengaruhi keamanan penerapan algoritma ini[8].

Algoritma Blowfish terdiri atas dua bagian, yaitu ekspansi kunci dan enkripsi data (Schneier, 1996).

a. Ekspansi kunci (Key-expansion)

Berfungsi merubah kunci (minimum 32-bit, maksimum 448-bit) menjadi beberapa array subkunci (subkey) dengan total 4168 byte (18x32-bit untuk P-array dan 4x256x32-bit untuk S-box sehingga totalnya 33344 bit atau 4168 byte). Kunci disimpan dalam K-array [9]:

K1, K2, ... Kj $1 \leq j \leq 14$

Kunci-kunci ini yang dibangkitkan (generate) dengan menggunakan subkunci yang harus dihitung terlebih dahulu sebelum enkripsi atau dekripsi data. Sub-sub kunci yang digunakan terdiri dari : P-array yang terdiri dari 18 buah 32-bit subkunci,

P1, P2, ..., P18

S-box yang terdiri dari 4 buah 32-bit, masing-masing memiliki 256 entri :

S1,0, S1,1, ..., S1,255

S2,0, S2,1, ..., S2,255

S3,0, S3,1, ..., S3,255

S4,0, S4,1, ..., S4,255

Langkah-langkah perhitungan atau pembangkitan subkunci tersebut adalah sebagai berikut:

1. Inisialisasi P-array yang pertama dan juga empat S-box, berututan, dengan string yang telah pasti. String tersebut terdiri dari digit-digit heksadesimal dari phi, tidak termasuk angka tiga di awal. Contoh :

   P1= 0x243f6a8

   P2= 0x85a308d3

   P3= 0x13198a2e

   P4= 0x03707344

   dan seterusnya sampai S-box yang terakhir (daftar heksadesimal digit dari phi untuk P-array dan Sbox bisa lihat Lampiran).
2. XOR-kan P1 dengan 32-bit awal kunci, XOR-kan P2 dengan 32-bit berikutnya dari kunci, dan seterusnya untuk semua bit kunci. Ulangi siklus seluruh bit kunci secara berurutan sampai seluruh P-array ter-XOR-kan dengan bit-bit kunci. Atau jika disimbolkan : P1 = P1 $\oplus$ K1, P2 = P2 $\oplus$ K2, P3 = P3 $\oplus$ K3, . . . P14 = P14 $\oplus$ K14, P15 = P15 $\oplus$ K1, . . . P18 = P18 $\oplus$ K4.

   Keterangan : $\oplus$ adalah simbol untuk XOR.
3. Enkripsikan string yang seluruhnya nol (all-zero string) dengan algoritma Blowfish, menggunakan subkunci yang telah dideskripsikan pada langkah 1 dan 2.
4. Gantikan P1 dan P2 dengan keluaran dari langkah 3.

5. Enkripsikan keluaran langkah 3 menggunakan algoritma Blowfish dengan subkunci yang telah dimodifikasi.
6. Gantikan P3 dan P4 dengan keluaran dari langkah 5.
7. Lanjutkan langkah-langkah di atas, gantikan seluruh elemen P-array dan kemudian keempat S-box secara berurutan, dengan hasil keluaran algoritma Blowfish yang terus-menerus berubah.

Total keseluruhan, terdapat 521 iterasi untuk menghasilkan subkunci-subkunci dan membutuhkan memori sebesar 4KB

b. Enkripsi Data

Terdiri dari iterasi fungsi sederhana (Feistel Network) sebanyak 16 kali putaran (iterasi), masukannya adalah 64-bit elemen data X. Setiap putaran terdiri dari permutasi kunci-dependent dan substitusi kunci- dan datadependent. Semua operasi adalah penambahan (addition) dan XOR pada variabel 32-bit. Operasi tambahan lainnya hanyalah empat penelusuran tabel array berindeks untuk setiap putaran. Langkahnya adalah seperti berikut.

1. Bagi X menjadi dua bagian yang masing-masing terdiri dari 32-bit: XL, XR.
2. Lakukan langkah berikut

   For i = 1 to 16:

   XL = XL $\oplus$ Pi

   XR = F(XL) $\oplus$ XR

   Tukar XL dan XR
3. Setelah iterasi ke-16, tukar XL dan XR lagi untuk melakukan membatalkan pertukaran terakhir.

4. Lalu lakukan

XR = XR ⊕ P17

XL = XL ⊕ P18

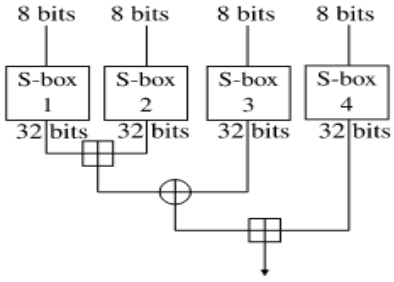
5. Terakhir, gabungkan kembali XL dan XR untuk mendapatkan cipherteks.

Untuk lebih jelasnya, gambaran tahapan pada jaringan feistel yang digunakan Blowfish adalah seperti pada Gambar 5.

Pada langkah kedua, telah dituliskan mengenai penggunaan fungsi F. Fungsi F adalah: bagi XL menjadi empat bagian 8-bit: a,b,c dan d.

F(XL) = ((S1,a + S2,b mod 232) XOR S3,c) + S4,d mod 232 ……(2.1)

Agar dapat lebih memahami fungsi F, tahapannya dapat dilihat pada Gambar 4.



Gambar 4. Fungsi F [9][10]

Dekripsi sama persis dengan enkripsi, kecuali bahwa P1, P2,…, P18 digunakan pada urutan yang berbalik (reverse). Algoritmanya dapat dinyatakan sebagai berikut:

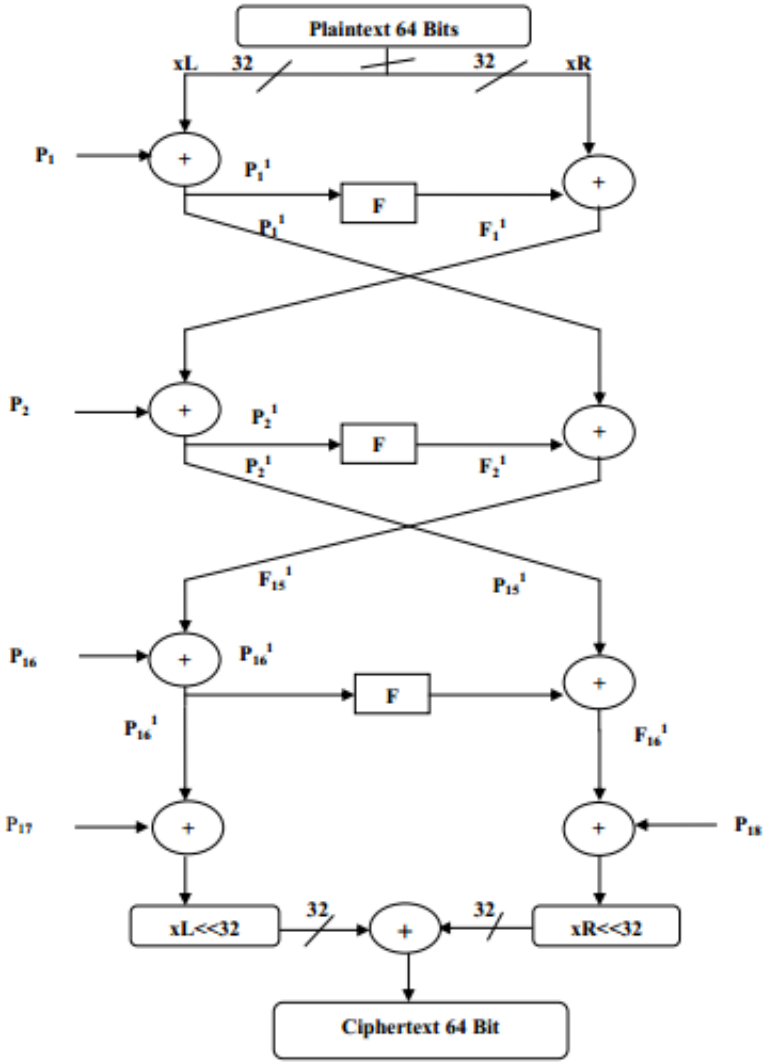
for i = 1 to 16 do

XRi = XLi-1 ⊕ P19-i

XLi = F[XRi] ⊕ XRi-1;

XL17 = XR16 ⊕ P1;

XR17 = XL16 ⊕ P2;



Gambar 5. Blok Diagram Algoritma Enkripsi Blowfish [9][10]

3. METODOLOGI PENELITIAN

3.1 Perancangan Antarmuka (Interface)

Perancangan Antarmuka Perancangan interface adalah bagian yang penting dalam aplikasi karena yang pertama kali dilihat ketika aplikasi dijalankan adalah interface aplikasi. Perancangan antarmuka sendiri terdiri dari perancangan antarmuka menu utama, perancangan antarmuka about dan perancangan antarmuka help. Perancangan antarmuka sendiri menggunakan bahasa XML.

3.2. Pembuatan Kelas Encryption

Kelas encryption ini adalah kelas yang digunakan untuk melakukan proses enkripsi dan dekripsi file. Pada pembuatan program ini, digunakan software Eclipse dengan menggunakan Java. Tampilan pada aplikasi OS Android diatur oleh file XML yang terdapat pada folder res/layout. Tabel 1 merupakan daftar tampilan layout yang telah dibuat.

Tabel.1 Nama File *Layout* Beserta Fungsinya

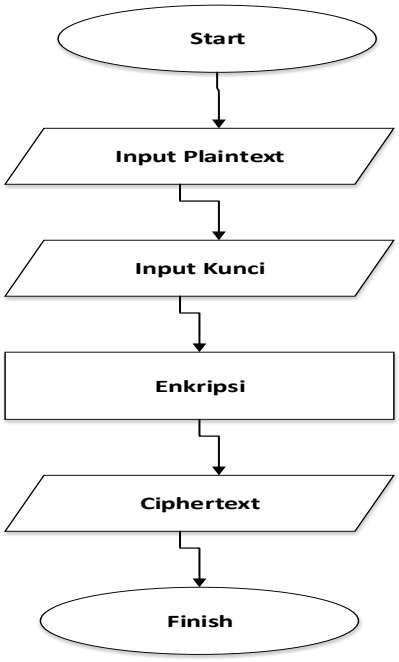
Nama File	Fungsi
activity_inbox_sms.xml	Mengatur tampilan pesan yang masuk
activity_send_sms.xml	Mengatur tampilan pesan yang dikirim
activity_smskripto.xml	Mengatur tampilan utama yang menampilkan fitur-fitur
Listsms.xml	Mengatur tampilan daftar list pesan

3.3 Proses Enkripsi File

Pada tahap ini merancang program untuk mengenkripsi text dengan algoritma Blowfish. Diagram blok untuk proses enkripsi text seperti pada Gambar 6.

Gambar 6. Diagram Blok Proses Enkripsi File

Gambar 7 memperlihatkan flowchart proses enkripsi text secara keseluruhan. Untuk melakukan proses enkripsi text hal pertama yang dilakukan adalah input plainteks berupa text.



Gambar 7. Flowchart Proses Enkripsi

Kunci yang digunakan untuk proses enkripsi text bisa berupa gabungan angka, huruf dan karakter khusus sesuai keinginan dari penggunanya. Blowfish sendiri menggunakan kunci simetris dimana kunci untuk enkripsi dan dekripsi sama. Setelah proses enkripsi text berhasil maka hasil outputnya berupa cipherteks yang sudah tidak dapat dimengerti maknanya.

3.4 Proses Dekripsi File

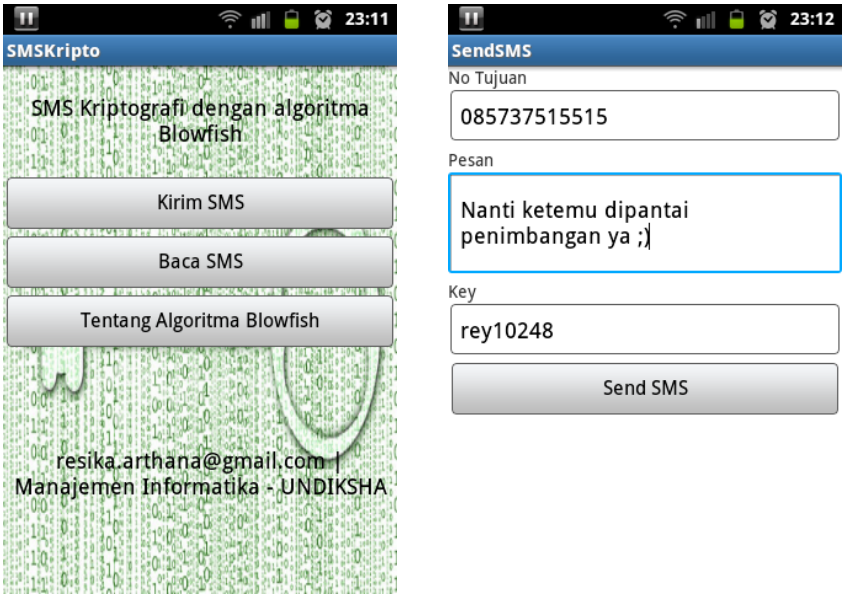
Pada tahap ini merancang program untuk mendekripsi file menggunakan algoritma Blowfish. Diagram blok sistem untuk proses dekripsi file diperlihatkan pada Gambar 8.

Gambar 8. Diagram Blok Proses Dekripsi File

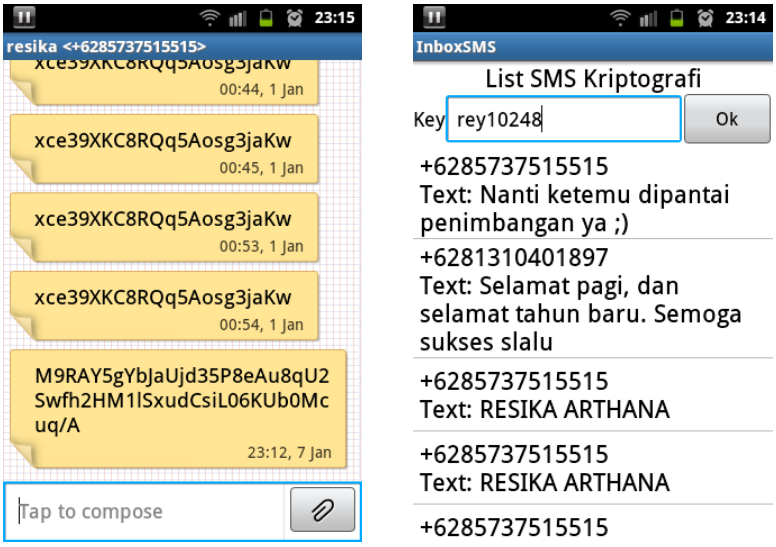
Kunci yang digunakan untuk proses enkripsi text bisa berupa gabungan angka, huruf dan karakter khusus sesuai keinginan dari penggunanya. *Blowfish* sendiri menggunakan kunci simetris dimana kunci untuk enkripsi dan dekripsi sama. Setelah proses dekripsi text berhasil maka hasil *output*nya berupa plainteks yang bisa dimengerti maknanya.

4. IMPLEMENTASI

Pada tahap ini, akan ditampilkan hasil implementasi dari penyusunan dan pengembangan kode program yang telah dibahas pada sub bab sebelumnya. Hasil implementasi yang akan ditampilkan berupa halaman anatar muka sistem.



Gambar 9. Tampilan Menu dan Kirim Pesan



Gambar 10. Tampilan isi sms tidak menggunakan dan yang menggunakan aplikasi

5. KESIMPULAN

5.1 Kesimpulan

Berdasarkan penjelasan dan pembahasan yang telah diuraikan pada bab – bab sebelumnya dan hingga tahap implementasi program. Maka dapat diambil kesimpulan bahwa :

1) Perancangan aplikasi enkripsi sms dengan algoritma blowfish dapat dilakukan dengan membuat tampilan aplikasi. Selanjutnya, pembuatan kode program untuk mengirim dan menerima pesan serta proses enkripsi dan dekripsi. Perancangan pada platform android dapat memanfaatkan aplikasi IDE seperti Eclips dengan bantuan ADT (Android Delopment Tool) dan Android SDK.
2) Implementasi blowfish memamahami alur dari algoritma tersebut, lalu mengimplementasikan pada bahasa Java. Algoritma Blowfish juga sudah ada pada JDK (Java Development Kit), untuk mengimplementasikan nya dapat menggunakan class SecretKeySpec yang digunkan untuk membangun upa kunci dan class Chiper yang digunakan untuk melakukan enkripsi dan dekripsi pesan.
3) pengembangan selanjutnya agar dapat meningkatkan fungsional dan manfaat aplikasi ini. Beberapa saran dari penulis untuk pengembangan aplikasi ini yaitu:
   - Menambahkan algortima kriptografi yang lain agar dapat menjadi pilihan enkripsi. Sedangkan untuk prioritas kekuatan keamanan disarankan agar menggunakan algoritma kriptografi yang lebih kuat atau terbaru.

- Aplikasi ini masih menggunakan database pesan standar diharapkan dapat membangun database mandiri agar dapat menyimpan pesan masuk maupun keluar.

- Aplikasi ini masih sederhana, disarankan menambahkan fitur seperti simpan kunci ataupun kirim kunci otomatis agar mempermudah bagi penerima pesan.

DAFTAR PUSAKA

[1] P. Traynor, W. Enck, P. McDaniel and T. La Porta, *Mitigating Attacks on Open Functionality in SMS-Capable Cellular Networks*, IEEE/ACM Transactions on Networking, 17(1):40-2009.

[2] Rahayu,Tri Puji, Yakub, Limiady,Irwan, *Aplikasi Enkripsi Pesan Teks (SMS) Pada Perangkat Handphone Dengan Algoritma Caesar Cipher*, Seminar Nasional Teknologi Informasi dan Komunikasi 2012 (SENTIKA 2012) ISSN: 2089-9815, 10 Maret 2012.

[3] Pangestu, Tegar Aji, *Implementasi Algoritma Rijndael pada Aplikasi Android Pengirim Short Message Service (SMS) Terenkripsi*, Makalah IF2120 Matematika Diskrit, Sem. I Tahun 2013/2014.

[4] E.A.Shanty, *Implementasi Algoritma Kriptografi Blowfish Untuk Keamanan Dokumen Pada Microsoft Office*, J. Chem. Inf. Model., vol. 53, no. 9, pp. 1689–1699, 2013.

[5] A. Rahman, T. Maskes, *Implementasi Algoritma Serpent Untuk Enkripsi Dan Dekripsi Data File Pada Ponsel*, J. Jur. Tek. Inform. STMIK GI MDP, 2013.

[6] R. Munir, Kriptografi. Bandung, Indonesia: Penerbit Informatika Bandung, 2006.

[7] R. Munir, "Diktat Kuliah IF5054 Kriptografi," Bandung, Indonesia, 2004.

[8] Tanjyot Aurora, Parul Arora , " Blowfish Algorithm " , IJCSCE , ISSN 2319-7080 , NCRAET , 2013.

[9] Christina L , Joe Irudayaraj V S, Optimized Blowfish Encryption Technique, International Journal of Innovative Research in Computerand Communication Engineering , ISSN: 2320-9801 , Vol. 2, Issue 7, July 2014.

[10] B.Schneier, The Blowfish Encryption Algorithm. July 22, 2009.