PENERAPAN KRIPTOGRAFI DENGAN ALGORITMA DES (*DATA ENCRYPTION STANDARD*) DAN BLOWFISH PADA SMS BERBASIS ANDROID

SKRIPSI

Disusun Oleh:

Danang Nur Fadlilah Akbar

12.18.178

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2016

LEMBAR PERSETUJUAN DAN PENGESAHAN

PENERAPAN KRIPTOGRAFI DENGAN ALGORITMA DES *(DATA ENCRYPTION STANDARD)* DAN BLOWFISH PADA SMS BERBASIS ANDROID

SKRIPSI

Disusun dan Diajukan untuk melengkapi dan memenuhi persyaratan guna mencapai Gelar Sarjana Komputer Strata Satu (S-1)

Disusun Oleh:

Danang Nur Fadlilah Akbar

12.18.178

Diperiksa dan Disetujui,

Dosen Pembimbing I	Dosen Pembimbing II
Joseph Dedy Irawan, ST.,MT.	**Nurlaily Vendyansyah, ST.**
NIP. 197404162005021002	**NIP.P -**

Mengetahui

Ketua Program Studi Teknik Informatika S-1

Joseph Dedy Irawan, ST., MT.

NIP. 197404162005011002

PROGRAM STUDI TEKNIK INFORMATIKA S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2016

PERNYATAAN KEASLIAN SKRIPSI

Yang bertanda tangan dibawah ini :

Nama : Danang Nur Fadlilah Akbar

NIM : 12.18.178

Jurusan : Teknik Informatika S-1

Fakultas : Teknologi Industri

Institut Teknologi Nasional Malang

Menyatakan dengan sesungguhnya bahwa skripsi saya yang berjudul :

"PENERAPAN KRIPTOGRAFI DENGAN ALGORITMA DES *(DATA ECRYPTION STANDARD)* DAN BLOWFISH PADA SMS BERBASIS ANDROID"

Adalah skripsi saya sendiri bukan duplikat serta mengutip atau menyadur seluruhnya karya orang lain kecuali dari sumber aslinya.

Malang, 24 Juli 2016

Yang membuat pernyataan

Danang Nur Fadlilah Akbar

PENERAPAN KRIPTOGRAFI DENGAN ALGORITMA DES *(DATA ENCRYPTION STANDARD)* DAN BLOWFISH PADA SMS BERBASIS ANDROID

Danang Nur Fadlilah Akbar (1218178)
Program Studi Teknik Informatika S-1, Fakultas Teknologi Industri
Institut Teknologi Nasional Malang, Jalan Karanglo km 2 Malang, Indonesia
E-mail : dhanank4kbar@gmail.com

ABSTRAK

Layanan SMS (Short Message Service) sebagai layanan pertukaran informasi atau pesan pendek menjadi komunikasi favorit karena saat ini semua telepon genggam memiliki layanan ini. 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. SMS yang bekerja pada jaringan nirkabel yang memungkinkan terjadinya pencurian isi pesan SMS ketika dalam proses transmisi dari pengirim ke penerima, kasus ini sering disebut dengan SMS interception.

Pada penelitian (Suristi:2010) menggunakan algoritma blowfish untuk pengamanan file. Pada penelitian ini digunakan algoritma DES dan Blowfish untuk pengamanan pesan SMS. Aplikasi ini merupakan aplikasi kriptografi simetri untuk pengamanan pesan SMS berbasis android yang dibangun dengan menggunakan android studio. Pada penelitian ini menggunakan algoritma DES yang mengenkripsikan 64 bit teks-asli menjadi 64 bit teks kode dengan 56 bit kunci. Kunci internal dibangkitkan dari kunci eksternal yang panjangnya 64 bit, dan Blowfish yang menerapkan teknik kunci berukuran sembarang. Ukuran kunci yang dapat diterima oleh blowfish adalah antara 32 bit hingga 448 bit, dengan ukuran default sebesar 128 bit.

Berdasarkan pengujian yang sudah dilakukan pada anaroid versi 4.2 jellybean, android versi 5.1 lollipop, dan android versi 6.1 marshmallow didapatkan hasil aplikasi dapat berjalan 100% dengan baik pada perangkat mobile berbasis android tersebut. Berdasarkan pengujian terhadap 6 responden didapatkan hasil bahwa aplikasi 100% bermanfaat dalam mengamankan pesan SMS.

Kata kunci: *Kriptografi, Enkripsi, Dekripsi, SMS, Android, DES, Blowfish*

KATA PENGANTAR

Dengan memanjatkan puji syukur kami panjatkan kehadirat yang Maha Agung Allah SWT yang selalu memberikan Rahmat dan HidayahNya yang telah dilimpahkan, sehingga penulis dapat menyelesaikan Skripsi dengan judul **"Penerapan Kriptografi Dengan Algoritma DES *(Data Encryption Standard)* Dan Blowfish Pada SMS Berbasis Android"** dengan lancar tanpa menemukan hambatan yang berarti. Skripsi ini merupakan persyaratan kelulusan di program Studi Teknik Informatika S-1 Institut Teknologi Nasional Malang dan untuk mencapai gelar Sarjana Komputer.

Keberhasilan penyelesaian laporan Skripsi ini tidak lepas dari dukungan dan bantuan berbagai pihak. Oleh karena itu pada kesempatan ini dengan segala kerendahan hati penyusun mengucapkan terima kasih kepada :

1. Allah SWT yang telah memberikan kesehatan bagi penyusun sehingga dapat mengerjakan laporan Skripsi.
2. Ayah dan Mama tercinta, karena selalu berdoa yang terbaik dan selalu memberikan dorongan baik secara moral maupun materil untuk menyelesaikan Skripsi ini.
3. Bapak Dr. Ir. Lalu Mulyadi, MT, selaku Rektor Institut Teknologi Nasional Malang.
4. Bapak Ir. Anang Subardi, MT, selaku Dekan Fakultas Teknologi Indrustri Institut Teknologi Nasional Malang.
5. Bapak Joseph Dedy Irawan, ST, MT, selaku Ketua Jurusan Teknik Informatika S-1 Institut Teknologi Nasional Malang.
6. Sonny Prasetio, ST, MT, selaku Sekertaris Program Studi Teknik Informatika S-1 Institut Teknologi Nasional Malang.
7. Bapak Joseph Dedy Irawan, ST, MT, selaku Dosen Pembimbing I Skripsi Jurusan Teknik Informatika S-1 Institut Teknologi Nasional Malang.
8. Ibu Nurlaily Vendyansyah, ST, selaku Dosen Pembimbing II Skripsi Jurusan Teknik Informatika S-1 Institut Teknologi Nasional Malang.
9. Mega Pambudi Ningtyas yang selalu setia mendoakan dan memberikan semangat dalam proses penyusunan Skripsi ini.

10. Sahabat sahabati tercinta yang telah memberikan banyak gagasan ide dan masukan.
11. Sanak famili serta keluarga yang selalu memberikan do'a restu, dorongan dan semangat.
12. Teman – teman dan semua yang tak mungkin disebutkan satu per satu yang telah membantu dalam penyelesaian penyusunan Skripsi ini.

Penulis telah berusaha semaksimal mungkin dan menyadari sepenuhnya akan keterbatasan pengetahuan dalam menyelesaikan laporan Skripsi ini. Untuk itu penyusun mengharapkan kritik dan saran yang membangun dari pembaca demi kesempurnaan laporan Skripsi ini.

Akhir kata penyusun mohon maaf yang sebesar-besarnya bilamana dalam penyusunan laporan Skripsi ini terdapat kekurangan serta kesalahan dalam penulisan. Semoga laporan Skripsi ini bermanfaat bagi pembaca.

Malang, 24 Juli 2016

Penulis

DAFTAR ISI

HALAMAN JUDUL i
LEMBAR PERSETUJUAN DAN PENGESAHAN ii
PERNYATAAN KEASLIAN SKRIPSI iii
ABSTRAK iv
KATA PENGANTAR v
DAFTAR ISI viii
DAFTAR GAMBAR x
DAFTAR TABEL xii

BAB I. PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 3
1.4 Tujuan 3
1.5 Manfaat 3
1.6 Metode Penelitian 4
1.7 Sistematika Penulisan 4

BAB II. LANDASAN TEORI 6
2.1 *Kriptografi* 6
2.1.1 Pengertian *Kriptografi* 6
2.1.2 Algoritma *Kriptografi* 9
2.1.3 Jenis Algoritma *Kriptografi* 9
2.2 DES *(Data Encryption Standard)* 12
2.2.1 Skema Global DES 12
2.2.2 Algoritma DES 14
2.2.3 Permutasi Awal 17
2.2.4 Pembangkitan Kunci Internal 18
2.2.5 Enkripsi Dan Dekripsi Algoritma DES 21
2.2.6 Permutasi Terakhir 26

2.3 Blowfish 26
2.3.1 Enkripsi Dan Dekripsi Algoritma Blowfish 28
2.4 Android 32
2.5 SMS *(Short Message Service)* 34
2.5.1 Cara Kerja SMS 35

BAB III. ANALISIS DAN PERANCANGAN 38
3.1 Analisis 38
3.1.1 Sistem Saat Ini 38
3.1.2 Contoh Kasus Dan Permasalahan 38
3.1.3 Contoh Kasus Dan Permasalahan 38
3.1.4 Sistem Yang Akan Dibangun 48
3.1.5 Struktur Umum Sistem 48
3.2 Perancangan Sistem 49
3.2.1 Desain Sistem 49
3.2.2 Struktur Menu 50
3.2.3 *Flowchart* 51
3.2.3.1 *Flowchart* Program Aplikasi Kriptografi 51
3.2.3.2 *Flowchart* Prosedur Enkripsi dan Dekripsi DES 52
3.2.3.3 *Flowchart* Prosedur Enkripsi dan Dekripsi Blowfish 56
3.2.3.4 *Flowchart* Prosedur Enkripsi dan Dekripsi Gabungan 59
3.3.4 Desain *Layout* 64

BAB IV. IMPLEMENTASI DAN PENGUJIAN 64
4.1 Implementasi Hasil 64
4.1.1 Tampilan Menu Utama 64
4.1.2 Tampilan Menu Tulis Pesan 65
4.1.3 Tampilan Menu Baca Pesan 65
4.1.4 Tampilan Menu Bantuan 66
4.1.5 Tampilan Menu Tentang 66
4.1.6 Tampilan *Alert Dialog* 67
4.1.7 Tampilan Hasil Enkripsi 67

DAFTAR GAMBAR

Gambar 2.1 *Kriptografi* Simetri *(Symetric Cryptography)* 10
Gambar 2.2 *Kriptografi* Asimetri *(Asymetric Cryptography)* 11
Gambar 2.3 Skema global DES 13
Gambar 2.4 Skema enkripsi DES 16
Gambar 2.5 Diagram alir pembangkitan kunci-kunci internal DES 21
Gambar 2.6 Diagram alir fungsi f algoritma DES 22
Gambar 2.7 Algoritma blowfish 30
Gambar 2.8 Fungsi f dalam blowfish 30
Gambar 2.9 Flowchart f fungsi 31
Gambar 2.10 Arsitektur dasar jaringan SMS 35
Gambar 3.1 Desain sistem aplikasi *kriptografi* 49
Gambar 3.2 Struktur menu aplikasi *kriptografi* 50
Gambar 3.3 Flowchart aplikasi *kriptografi* 51
Gambar 3.4 *Flowchart* prosedur enkripsi DES 53
Gambar 3.5 *Flowchart* prosedur dekripsi DES 55
Gambar 3.6 *Flowchart* prosedur enkripsi blowfish 56
Gambar 3.7 *Flowchart* prosedur dekripsi blowfish 58
Gambar 3.8 *Flowchart* enkripsi gabungan DES dan blowfish 59
Gambar 3.9 *Flowchart* dekripsi gabungan DES dan blowfish 56
Gambar 3.10 Halaman Utama aplikasi kriptografi 61
Gambar 3.11 Menu Tulis pesan aplikasi kriptografi 61
Gambar 3.12 Menu Kotak masuk aplikasi kriptografi 62
Gambar 3.13 Menu Bantuan aplikasi kriptografi 63
Gambar 3.14 Menu Tentang aplikasi kriptografi 63
Gambar 4.1 Tampilan menu utama 64
Gambar 4.2 Tampilan menu tulis pesan 65
Gambar 4.3 Tampilan menu baca pesan 65
Gambar 4.4 Tampilan menu bantuan 66
Gambar 4.5 Tampilan menu tentang 66

Gambar 4.6 Tampilan alert dialog.....67
Gambar 4.7 Tampilan menu hasil enkripsi 67
Gambar 4.8 Tampilan menu hasil dekripsi 68

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi dalam media komunikasi berkembang cepat. Dari komunikasi suara sampai komunikasi data. Penggunaan teknologi telepon genggam *(handphone)* sebagai alat telekomunikasi pada saat ini telah 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 komputer, 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 mereka. Android berkembang pesat karena mempunyai platform yang sangat lengkap baik dalam system operasi, aplikasi dan tool pengembangannya, market aplikasi serta mendapat dukungan yang sangat tinggi dari komunitas Open Source di dunia (Safaat, 2012).

Meskipun Android memiliki fitur yang lengkap, namun layanan SMS *(Short Message Service)* sebagai layanan pertukaran informasi atau pesan pendek menjadi 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. Celah keamanan terbesar pada layanan komunikasi SMS adalah pada saat SMS tersebut sedang dikirim melalui jaringan SMS tersebut. SMS bekerja pada jaringan nirkabel yang memungkinkan terjadinya pencurian isi pesan SMS ketika dalam proses transmisi dari pengirim ke penerima, kasus ini disebut SMS interception.

Dengan adanya beberapa keterangan diatas maka dibutuhkan sebuah sistem keamanan pada layanan SMS yang mampu menjaga integritas dan

keamanan isi pesan bagi setiap orang terutama pada kalangan pemerintah yang membutuhkan privasi untuk menutupi celah keamanan SMS terutama untuk SMS interception dengan menggunakan teknik kriptografi yang sudah ada. Kriptografi adalah seni untuk mengamankan informasi dengan menggunakan teknik penyandian. Proses penyandian informasi asli (plainteks) yang menghasilkan informasi yang tersandikan (chiperteks) disebut enkripsi, sedangkan proses menguraikan chiperteks menjadi informasi asli disebut dekripsi. Dalam penelitian ini menggunakan kriptografi dengan metode algoritma DES *(Data Encryption Standard)* dan algoritma blowfish.

Adapun penggunaan DES dan bukan AES *(Advance Ecryption Standard)* dikarenakan DES lebih termasuk sistem kriptografi simetri dan tergolong jenis blok kode sedangkan AES merupakan kriptografi asimetri. DES beroperasi pada ukuran blok 64 bit. DES mengenkripsikan 64 bit teks-asli menjadi 64 bit teks kode dengan 56 bit kunci internal *(internal key)* atau upa-kunci *(subkey)*. Kunci internal dibangkitkan dari kunci eksternal *(external key)* yang panjangnya 64 bit. Sedangkan Blowfish adalah algoritma kriptografi kunci simetri blok kode dengan panjang blok tetap 64 bit. Blowfish menerapkan teknik kunci berukuran sembarang. Ukuran kunci yang dapat diterima oleh blowfish adalah antara 32 bit hingga 448 bit, dengan ukuran default sebesar 128 bit. Blowfish memanfaatkan teknik pemanipulasian bit dan teknik pemutarn ulang dan pergiliran kunci yang dilakukan sebanyak 16 kali. Algoritma utama terbagi menjadi dua subalgoritma utama, yaitu bagian ekspansi kunci dan bagian enkripsi-dekripsi data.

Dengan demikian penulis ingin membuat suatu sistem **"Penerapan Kriptografi Dengan Algoritma DES *(Data Encryption Standard)* Dan Blowfish Pada SMS Berbasis Android".**

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah diatas, dapat dirumuskan permasalahan yang akan dibahas adalah bagaimana menerapan kriptografi dengan algoritma DES *(data encryption standard)* dan blowfish pada sms berbasis android?

1.3 Batasan Masalah

Dalam penyusunan Skripsi ini, dak menyimpang dari permasalahan yang ditetapkan, maka penyusun membatasi masalah Skripsi ini pada:

1. Algoritma Kriptografi yang digunakan menggunakan algoritma DES dan Blowfish.
2. Media pengamanan pesan menggunakan *mobile* berbasis android.
3. Pesan yang disisipkan atau disembunyikan hanya terbatas pada 160 karakter dan bertipe pesan teks *(.txt)*.
4. Minimum *operating sistem* yang dibutuhkan untuk aplikasi ini adalah versi android 4.2 atau jellybean.

1.4 Tujuan

Berdasarkan rumusan masalah diatas, maka dalam Skripsi ini tujuan yang akan dicapai adalah:

"Untuk mengetahui penerapan kriptografi dengan algoritma DES *(data encryption standard)* dan blowfish pada sms berbasis android"

1.5 Manfaat

Manfaat yang dapat diambil dari Aplikasi ini adalah:

1. Bagi penulis, sebagai sarana menambah ilmu dan pengetahuan dalam bidang sistem pengamanan pesan berbasis android.
2. Bagi pengguna aplikasi, membantu user atau pengguna dalam menjaga keamanan dan kerahasiaan pesan yang disampaikan melalui tulisan dari pengirim ke penerima.
3. Bagi masyarakat dan pihak lain, sebagai bahan referensi dan tambahan informasi untuk pengkajian topik yang berkaitan dengan masalah yang sama dengan penelitian ini dan digunakan sebagaimana mestinya.

1.6 Metode Penelitian

Adapun Metode Penelitian yang digunakan adalah sebagai berikut :

1. Studi Literatur

   Pada tahap ini dipelajari literatur dan perencanaa serta konsep awal untuk membentuk program yang akan dibuat yaitu didapat dari penelitian yang sudah ada, refrensi buku, internet maupun sumber-sumber yang lain.

2. Pengumpulan Data dan Analisis

   Pada tahap ini adalah proses perancangan dari sistem yang akan dibuat berdasarkan data yang sudah dikumpulkan serta analisa yang telah dilakukan pada tahap sebelumnya.

3. Analisa dan Perancangan Sistem

   Pada tahap ini adalah proses perancangan dari sistem yang di buat berdasarkan data yang sudah di kumpulkan serta analisa yang telak dilakukan pada tahap sebelumnya.

4. Pembuatan Program dan Implementasi

   Tahap ini selanjutnya adalah tahap pembuatan program dan impementasi dengan menggunakan bahasa pemrogaman.

5. Uji Coba Program

   Program selesai dibuat maka dilakukan pengujian program untuk mengetahui apakah program tersebut telah bekerja dengan benar dan sesuai dengan sistem yang dibuat.

1.7 Sistematika Penulisan

Dalam penyusunan skripsi ini agar lebih mudah dipahami maka dibuatlah suatu sistematika penulisan sebagai berikut:

BAB I : PENDAHULUAN

Bab ini berisi tentang penjelasan mengenai latar belakang, perumusan masalah, batasan masalah, tujuan, metode penelitian, dan sistematika penulisan.

BAB II : LANDASAN TEORI

Bab ini berisi tentang penjelasan mengenai tentang landasan teori-teori yang berkaitan dengan penelitian pembuatan aplikasi ini.

BAB III : ANALISIS DAN PERANCANGAN

Bab ini berisi tentang penjelasan perancangan aplikasi Kriptografi Dengan Algoritma DES *(Data Encryption Standard)* Dan Blowfish Pada SMS Berbasis Android yang akan dibuat.

BAB IV : IMPLEMENTASI DAN PENGUJIAN PROGRAM

Bab ini berisi tentang penjelasan implementasi terhadap aplikasi Kriptografi Dengan Algoritma DES *(Data Encryption Standard)* Dan Blowfish Pada SMS Berbasis Android dari awal hingga akhir, dan melakukan pengujian terhadap aplikasi tersebut.

BAB V : PENUTUP

Bab ini akan memuat tentang kesimpulan dari hasil penelitian yang telah dilakukan dan saran-saran yang berhubungan dengan skripsi yang dibuat untuk dijadikan pengembangan penelitian berikutnya.

BAB II

LANDASAN TEORI

2.1 Kriptografi

2.1.1 Pengertian Kriptografi

Kriptografi atau kriptologi; dari bahasa Yunani *crytos*, "tersembunyi, rahasia"; dan *graphein*, "menulis", atau logi, "ilmu") merupakan keahlian dan ilmu dari cara-cara untuk komunikasi aman pada kehadirannya di pihak ketiga. (wikipedia : 2010).

Secara singkat, sejarah perkembangan kriptografi menurut Kahate Atul (2013:15) dapat dijabarkan seperti berikut:

a) Tahun 1900 SM, pertama kali digunakannya teknik transformasi *cyptography* di "*tomb inscription*", merupakan penggunaan kriptografi yang pertama kali diketahui.
b) Tahun 475 SM, Sparta menggunakan kriptografi untuk komunikasi dan juga merancang alat untuk mengenkripsi (skytale) yang menghasilkan *transposition cipher*.
c) Tahun 350 SM, "Aenas The Tactician" mengeluarkan tulisan pertama mengenai keamanan komunikasi dan kriptografi.
d) Tahun 60 SM, Julius Caesar menjadi orang yang pertama kali yang diketahui menggunakan substitution *cipher*.
e) Tahun 1412, "Treatise" tertua yang diketahui dalam kriptanalis yang *diterbitkan* oleh Alkalkas Handi (Mesir).
f) Tahun 1917, Edward Hugh Hibern mengembangkan mesin motor yang pertama.
g) Tahun 1971, IBM mengembangkan teknik enkripsi Lucifer.
h) Tahun 1975, DES diumumkan (disetujui tahun 1977).
i) Tahun 1976, presentasi terbuka pertama tentang konsep *public key* oleh Diffie dan Helman.
j) Tahun 1977, Merkle mengembangkan algoritma *knapsack* dan memberikan hadiah $100 bagi yang dapat memecahkan kuncinya

(algoritma dengan satu kali pengulangan). Algoritma Rivest-Shamir-Aldeman (RSA) diumumkan kepada umum.

Sebagaimana banyak teknologi lainnya, selama bertahun-tahun kriptografi menjadi bidang khusus yang hanya dipelajari oleh pihak militer. Agen Keamanan Nasional Amerika (NSA = *National Security Agency*), Uni Soviet, Inggris, Perancis, Israel dan negara lainnya telah membelanjakan miliaran dolar untuk mengamankan komunikasi mereka dan pada saat yang bersamaan mereka pun berusaha memecahkan kode rahasia negara saingannya.

Namun pada kurun waktu 30 tahun terakhir, penelitian akademik di bidang kriptografi meledak dengan dahsyatnya. Kemajuan teknologi komputasi komputer menambah cepatnya perkembangan kriptografi. Sekarang, kriptografi bukan lagi monopoli militer, setiap individu berhak mengamankan komunikasinya tanpa kuatir dimata-matai oleh pihak lain. Setiap individu berhak melindungi komunikasi yang berisi rahasia keluarganya, bisnisnya, pekerjaannya, dan pendapat-pendapatnya. Hal ini tentu bertentangan dengan kebijakan pemerintah yang menginginkan agar setiap kegiatan rakyatnya dapat dikontrol, apalagi kalau dianggap melawan pemerintah. Bahkan di negara yang mengaku paling demokratis sedunia, Amerika, pemerintahnya justru mengontrol dengan sangat ketat kriptografi. Bila anda menginginkan produk kriptografi yang paling andal, jangan berharap anda dengan mudah mendapatkannya dari negeri ini. Pemerintah Amerika bahkan melarang ekspor produk kriptografi yang tidak mampu mereka pecahkan kode rahasianya. Sementara untuk komunikasi dalam negeri, mereka mengizinkan penggunaan produk kriptografi yang lebih kuat, karena pemerintah Amerika dapat melakukan penyadapan komunikasi dengan mudah tanpa harus memecahkan kriptografinya. Pemerintah Amerika menganggap produk kriptografi sebagai amunisi semacam tank, rudal, pesawat pembom maupun kapal-kapal perangnya.

Menurut Kahate Atul (2013:32) "*Cyptography is the art of security by encoding messages to make them non-readable*", yang artinya kriptografi adalah seni keamanan dengan pengkodean pesan agar tidak bisa dibaca. Ilmu kriptogarfi adalah ilmu yang mempelajari tentan penyembunyian huruf atau tulisan sehingga membat tulisan tersebut tidak dapat dibaca oleh orang yang tidak berkepentingan

(Sofwan Aghus, 2006:23). Request for Comments (RFC) mengemuanan bahwa kriptografi merupakan ilmu matematika yang berhubungan dengan transformasi data untuk membuat artinya tidak dapat dipahami (untuk menyembunyikan maknanya), mencegahnya dari perubahan tanpa izin, atau mencegahnya dari penggunaan yang tidak sah. Jika transformasinya dapat dikembalikan, kriptografi juga bisa diartikan sebagai proses mengubah kembali data yang terenkripsi menjadi bentuk yang dapat dipahami.

Dalam menjaga kerahasiaan data, kriptografi mentransformasikan data jelas (*plaintext*) ke dalam bentuk data sandi (*ciphertext*) yang tidak dapat dikenali. Ciphertext inilah yang kemudian dikirimkan oleh pengirim (*sender*) kepada penerima (*receiver*). Setelah sampai di penerima, *ciphertext* tersebut ditranformasikan kembali ke dalam bentuk *plaintext* agar dapat dikenali.

Dalam arti lain, *cryptography* adalah seni dan ilmu dalam mengamankan pesan. Dalam dunia kriptografi, pesan disebut *plaintext* atau *cleartext*. Proses untuk menyamarkan pesan dengan cara sedemikian rupa untuk menyembunyikan isi aslinya disebut enkripsi. Pesan yang telah dienkripsi disebut *ciphertext*. Proses pengembalian sebuah *ciphertext* ke *plaintext* disebut dekripsi.

Cryptographer adalah orang yang mempraktekkan ilmu kriptografi, sedangkan *cryptoanalysts* adalah orang yang mempraktekkan kriptanalisis, seni dan ilmu dalam memecahkan *ciphertext*. Aturan fundamental kriptografi yaitu seseorang harus mengasumsikan bahwa seorang kriptoanalis menguasai algoritma umum enkripsi yang digunakan. Dengan kata lain, kriptanalis mengetahui cara kerja algoritma enkripsi. Jumlah usaha yang diperlukan untuk menemukan, menguji, dan memasang algoritma baru yang selalu berkompromi atau berfikir untuk berkompromi dengan algoritma lama, akan menyebabkan algoritma baru itu menjadi tidak berguna untuk menjaga kerahasiaan. Sistem kriptografi atau *Algoritma Kriptografi* adalah sebuah algoritma kriptografi ditambah semua kemungkinan *plaintext*, *ciphertext* dan kunci.

Berdasaran beberapa uraian diatas dapat di simpulkan bahwa Kriptografi didefinisikan sebagai ilmu dan pelajaran untuk tulisan rahasia dengan pertimbangan bahwa komunikasi dan data dapat dikodekan untuk mencegah dari mata-mata atau orang lain yang ingin mengetahui isinya, dengan menggunakan

kode-kode dan aturan-aturan tertentu dan metode lainnya sehingga hanya orang yang berhak yang dapat mengetahui isi pesan sebenarnya.

2.1.2 Algoritma Kriptografi

Algoritma menggambarkan sebuah prosedur komputasi yang terdiri dari variabel input dan menghasilkan output yang berhubungan (Oppliger, 2005). Algoritma kriptografi atau sering disebut dengan *cipher* adalah suatu fungsi matematis yang digunakan untuk melakukan enkripsi dan dekripsi (Schneier, 1996). Algoritma kriptografi ini bekerja dalam kombinasi dengan menggunakan kunci (*key*) seperti kata, nomor atau frase tertentu.

Bila keamanan algoritma bergantung pada kerahasian algoritma yang bekerja, maka algoritma tersebut dikatakan sebagai algoritma terbatas (terbatas kemampuannya). Algoritma terbatas mempunyai sejarah yang menarik. Algoritma terbatas biasanya digunakan oleh sekelompok orang untuk bertukar pesan satu sama lain. Mereka membuat suatu algoritma enkripsi dan algoritma enkripsi tersebut hanya diketahui oleh anggota kelompok itu saja. Tetapi, algoritma terbatas tidak cocok lagi saat ini, sebab setiap kali ada anggota kelompok keluar, maka algoritma kriptografi harus diganti lagi. Kerahasian algoritmanya menjadi titik kelemahan karena tidak mengijinkan adanya kontrol kualitas atau standarisasi.

2.1.3 Jenis Algoritma Kriptografi

Berdasarkan jenis kunci yang digunakannya, algoritma kriptografi dikelompokan menjadi dua bagian, yaitu algoritma simetris (algoritma konvensional) dan algoritma asimetris (algoritma kunci publik).

2.1.3.1 Kriptografi Simetri *(Symetric Cryptography)*

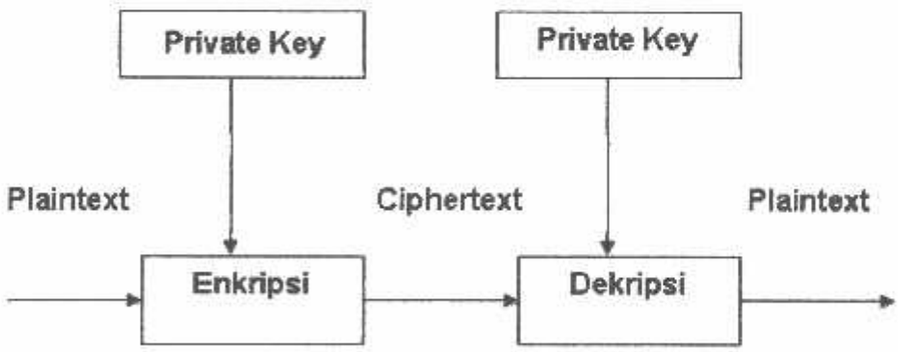
Algoritma simetris adalah algoritma kriptografi yang menggunakan kunci enkripsi yang sama dengan kunci dekripsinya. Istilah lain untuk kriptografi kunci-simetri adalah kriptografi kunci privat (*private-key cryptography*), kriptografi kunci rahasia (*secret-key cryptography*), atau kriptografi konvensional (*conventional cryptography*) (Ariyus Dony, 2008:44). Sistem kriptografi kunci-simetri (atau disingkat menjadi "kriptografi simetri" saja), mengasumsikan

pengirim dan penerima pesan sudah berbagi kunci yang sama sebelum bertukar pesan. Keamanan sistem kriptografi simetri terletak pada kerahasiaan kuncinya.

Pada sistem *kriptografi* simetri, kunci untuk proses enkripsi sama dengan kunci untuk proses dekripsi. Keamanan sistem *kriptografi* simetri terletak pada kerahasiaan kunci. Istilah lain untuk *kriptografi* simetri adalah *kriptografi* kunci privat *(private key cryptography)* atau *kriptografi* konvensional *(conventional cryptography)*.

Kelebihan algoritma simetris ini adalah proses enkripsi dan deskripsinya yang jauh lebih cepat dibandingkan dengan algoritma asimetris. Sedangkan kelemahan algoritma ini adalah permasalahan distribusi kunci (*key distribution*). Seperti yang telah dibahas, proses enkripsi dan deskripsi menggunakan kunci yang sama. Sehingga muncul persoalan menjaga kerahasian kunci, yaitu pada saat pengiriman kunci pada media yang tidak aman seperti internet. Tentunya jika kunci ini sampai hilang atau sudah dapat ditebak oleh orang lain (orang yang tidak berhak), maka kriptosistem ini sudah tidak aman lagi.

Kelemahan lain adalah masalah efisiensi jumlah kunci. Jika terdapat n user, maka diperlukan $n(n\text{-}1)/2$ kunci, sehingga untuk jumlah user yang sangat banyak, sistem ini tidak efisien lagi.



Gambar 2.1 Kriptografi Simetri (Symetric Cryptography)

Algoritma *kriptografi* simetri dapat dikelompokkan menjadi dua kategori antara lain:

1. Cipher aliran (stream cipher)

   Algoritma *kriptografi* beroperasi pada *plainteks/cipherteks* dalam bentuk bit tunggal yang dalam hal ini rangkaian bit dienkripsikan/didekripsikan bit per bit. Cipher aliran mengenkripsi satu bit setiap kali.

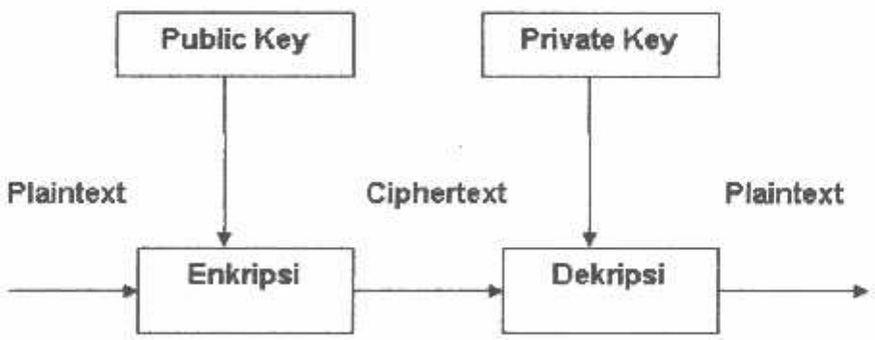
2. Cipher blok *(block cipher)*

   Algoritma *kriptografi* beroperasi pada *plainteks/cipherteks* dalam bentuk blok bit, yang dalam hal ini rangkaian bit dibagi menjadi blok-blok bit yang panjangnya sudah ditentukan sebelumnya. Cipher blok mengenkripsi satu blok bit setiap kali.

2.1.3.2 Kriptografi Asimetri (*Asymetric Cryptography*)

Algoritma asimetris atau kunci publik didesain sehingga kunci yang digunakan untuk enkripsi berbeda dengan kunci untuk dekripsi dimana kunci untuk enkripsi tidak rahasia dan dapat diketahui oleh siapapun (diumumkan ke publik), sementara kunci untuk dekripsi hanya diketahui oleh penerima pesan (rahasia) (Dony Ariyus, 2008:45).

Pada sistem *kriptografi* asimetri, kunci untuk proses enkripsi tidak sama dengan kunci untuk proses dekripsi. Istilah lain untuk *kriptografi* asimetri adalah kriptografi kunci publik (*public key cryptography*), sebab kunci untuk enkripsi tidak rahasia dan dapat diketahui oleh siapapun, sementara kunci untuk dekripsi hanya diketahui oleh penerima pesan.



Gambar 2.2 Kriptografi Asimetri (Asymetric Cryptography)

2.2 DES *(Data Encryption Standard)*

Data Encryption Standard (DES) adalah suatu blok cipher (salah satu bentuk enkripsi rahasia bersama) yang dipilih oleh National Bureau of Standar sebagai seorang pejabat Federal Information Processing Standard (FIPS) untuk Amerika Serikat pada tahun 1976 dan yang kemudian dinikmati secara luas gunakan internasional. Hal ini didasarkan pada algoritma kunci simetris yang menggunakan 56-bit key. Algoritma awalnya diklasifikasikan kontroversial dengan elemen desain, kunci yang relatif pendek panjang, dan kecurigaan tentang National Security Agency (NSA) backdoor. DES akibatnya datang di bawah pengawasan intens akademis yang memotivasi pemahaman modern dan blok cipher kriptoanalisis mereka (Peter dan Gnana, 2004:67).

DES merupakan salah satu algoritma kriptografi cipher block dengan ukuran blok 64 bit dan ukuran kuncinya 56 bit. Algoritma DES dibuat di IBM, dan merupakan modifikasi daripada algoritma terdahulu yang bernama Lucifer. Lucifer merupakan algoritma cipher block yang beroperasi pada blok masukan 64 bit dan kuncinya berukuran 28 bit. Pengurangan jumlah bit kunci pada DES dilakukan dengan alasan agar mekanisme algoritma ini bisa diimplementasikan dalam satu chip (Dony Ariyus (2008:112). DES atau DEA (Data Encryption Standard) merupakan algoritma kriptografi simetris yang paling umum digunakan saat ini. DES bermula dari Lucifer, enkripsi yang dikembangkan di IBM (Stiawan Deris, 2005:72)

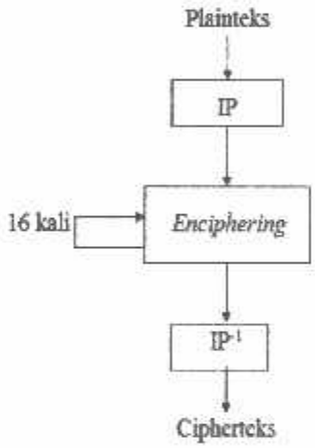
Berdasarkan uraian diatas dapat disimpulkan bawah DES (Data Encryption Standard) merupakan sebuah algoritma enkripsi sandi blok kunci simetrik dengan ukuran blok 64-bit dan ukuran kunci 56-bit.

2.2.1 Skema Global DES

Pada sekitar akhir tahun 1960, IBM melakukan riset pada bidang kriptografi yang pada akhirnya disebut Lucifer. Lucifer dijual pada tahun 1971 pada sebuah perusahaan di London. Lucifer merupakan algoritma berjenis Block Cipher yang artinya bahwa input maupun output dari algoritma tersebut merupakan 1 blok yang terdiri dari banyak bit seperti 64 bit atau 128 bit. Lucifer beroperasi pada blok input 64 bit dan menggunakan key sepanjang 128 bit. Lama kelamaan Lucifer semakin dikembangkan agar bisa lebih kebal terhadap

serangan analisis cypher tetapi panjang kuncinya dikurangi menjadi 56 bit dengan maksud supaya dapat masuk pada satu chip (Dony Ariyus (2008:113).

Di tempat yang lain, biro standar amerika sedang mencari-cari sebuah algoritma enkripsi untuk dijadikan sebagai standar nasional.IBM mencoba mendaftarkan algoritmanya dan di tahun 1977 algoritma tersebut dijadikan sebagai DES (Data Encryption Standard). Algoritma ini telah disetujui oleh National Bureau of Standard (NBS) setelah penilaian kekuatannya oleh National Security Agency (NSA) Amerika Serikat. DES termasuk ke dalam sistem kriptografi simetri dan tergolong jenis cipher blok. DES beroperasi pada ukuran blok 64 bit dan mengenkripsikan 64 bit plainteks menjadi 64 bit cipherteks dengan menggunakan 56 bit kunci internal (internal key) atau sub-kunci (subkey). Kunci internal dibangkitkan dari kunci eksternal (external key) yang panjangnya 64 bit. Skema global dari algoritma DES adalah sebagai berikut:



Gambar 2.3 Skema global DES

Keterangan:

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

2.2.2 Algoritma DES

Peter dan Gnana (2004:69) menyatakan bahwa di dalam proses enciphering, blok plainteks terbagi menjadi dua bagian, kiri (L) dan kanan (R), yang masing-masing panjangnya 32 bit. Kedua bagian ini masuk ke dalam 16 putaran DES. Pada setiap putaran i, blok R merupakan masukan untuk fungsi transformasi yang disebut f. Pada fungsi f, blok R dikombinasikan dengan kunci internal Ki. Keluaran dai fungsi f di-XOR-kan dengan blok L untuk mendapatkan blok R yang baru. Sedangkan blok L yang baru langsung diambil dari blok R sebelumnya. Ini adalah satu putaran DES.

Algoritma DES merupakan algoritma simetris yang paling banyak digunakan di dunia untuk proses enkripsi. DES bekerja pada bit atau bilangan biner yaitu 0 dan 1. Sebelum dienkripsi menggunakan algoritma DES , sebuah *plaintext* yang berupa bilangan hexadesimal (bilangan berbasis 16) akan di konversi terlebih dahulu kedalam 64 bit bilangan biner dengan masing-masing grup tersusun dari 4 bit. Sebagai contoh, bilangan hexadesimal "0" akan dikonversi kedalam bilangan biner "0000", bilangan hexadesimal "6" akan dikonversi kedalam bilangan biner "0110" dan bilangan hexadesimal "B" akan dikonversi kedalam bilangan biner "1011". Konversi bilangan hexadesimal ke bilangan biner selengkapnya disajikan dalam Tabel 2.1.

Begitu juga setelah proses enkripsi DES, hasil enkripsi yang berupa bilangan biner dikonversi kedalam bilangan hexadesimal. Jadi *plaintext* dan *ciphertext* yang merupakan input dan output dalam algoritma DES ini berupa bilangan hexadesimal.

Tabel 2.1 Konversi bilangan hexadesimal, biner dan desimal

Bilangan Hexadesimal	Bilangan Biner	Bilangan Desimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3

4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15

Satu putaran DES ini akan diulang sebanyak 16 kali putaran. Skema dari diagram alir enkripsi menggunakan algoritma DES secara lebih rinci disajikan pada Gambar 2.3.

di mana ⊕ merupakan *exclusive-or* dari dua. *f* adalah suatu fungsi dan K_1, K_2, . . ., K_{16} dengan panjang 48 dari perhitungan fungsi dari kunci *K*. (Sebenarnya K_i adalah permutasi dari *K*). K_1, K_2, . . ., K_{16} terdiri dari kunci skedul. Putaran pertama dari enkripsi tersebut ditunjukan oleh gambar dibawah ini:

3. Hasil echipering kemudian dipermutasi dengan matriks permutasi balik (invers initial permutation atau IP- 1) menjadi blok teks-kode. IP ' ke bitstring $R_{16}L_{16}$, memperoleh teks-kode y, kemudian y = IP $^{-1}$ ($R_{16}L_{16}$).

2.2.3 Permutasi Awal

Sebelum melakukan putaran pertama dalam proses enkripsi DES, terlebih dahulu dilakukan permutasi awal (IP) terhadap input yang berupa blok *plaintext* yang panjangnya 64 bit. Tujuan dari permutasi awal ini adalah untuk mengacak *plaintext* tersebut sehingga urutan bit-bit yang berada di dalamnya berubah. Pengacakan ini dilakukan menggunakan matriks permutasi awal sebagaimana dalam Tabel 2.2 sebagai berikut.

Tabel 2.2 *Initial Permutation (IP)*

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
63	55	47	39	31	23	15	7

Penerapan IP pada sebuah *plaintext* P adalah dengan memindahkan bit ke-58 dari P menjadi bit ke-1 pada IP, bit ke-50 dari P menjadi bit ke-2 pada IP, dan seterusnya sampai dengan bit ke-7 dari P menjadi bit ke-64 pada IP. Selanjutnya hasil permutasi dari *plaintext* P di enkripsi sebanyak 16 kali putaran menggunakan kunci internal yang berbeda.

2.2.4 Pembangkit Kunci Internal

Setiap blok *plaintext* mengalami 16 kali putaran pada proses enkripsi, untuk itu dibutuhkan kunci internal sebanyak 16 buah, yaitu K_1, K_2, ..., K_{16}. Kunci-kunci internal ini dapat dibangkitkan dari kunci eksternal yang diberikan oleh pengguna sebelum proses enkripsi atau bersamaan dengan proses enkripsi. Kunci eksternal yang panjangnya 64 bit atau 8 karakter menjadi input untuk permutasi menggunakan permutasi kompresi 1 (PC-1) sebagaimana dalam Tabel 2.3. Permutasi kompresi menggunakan Tabel 2.3 ini menghasilkan $K+$.

Tabel 2.3 Permutasi Kompresi 1

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

Dalam permutasi pada Tabel 2.3, setiap bit kelipatan delapan *(parity bit)* dari delapan byte kunci diabaikan, yaitu bit 8, 16, 24, 32, 40, 48, 56, dan 64. Hasil permutasi kompresi sepanjang 56 bit, sehingga dapat dikatakan kunci DES adalah 56 bit.

Input pertama dalam Tabel 2.3 adalah 57 yang artinya bit ke-57 pada kunci K menjadi bit pertama dalam permutasi $K+$. Bit ke 49 pada kunci K

menjadi bit kedua dalam permutasi K+, dan seterusnya bit ke-4 pada kunci K menjadi bit terakhir dalam permutasi K+.

Selanjutnya 56 bit $K+$ ini akan dibagi menjadi 2 blok yaitu blok kiri (C_0) dan blok kanan (D_0), dimana masing-masing panjangnya 28 bit yang tersimpan dalam C_0 dan D_0.

C_0 berisi bit-bit dari $K+$ pada posisi:

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

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

D_0 berisi bit-bit dari $K+$ pada posisi:

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

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

Setelah itu kedua blok tersebut digeser ke kiri (*left shift*) sepanjang satu bit untuk i=1, 2, 9 atau 16 dan digeser sepanjang dua bit untuk i yang lain sehingga diperoleh C_i dan D_i untuk $1 \leq i \leq 16$. Jumlah pergeseran pada setiap putaran ditunjukkan pada Tabel 2.4.

Tabel 2.4 Jumlah pergeseran bit pada setiap putaran

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

Misalkan (C_i,D_i) menyatakan penggabungan C_i dan D_i , (C_{i+1},D_{i+1}) dapat diperoleh dengan menggeser C_i dan D_i sepanjang satu atau dua bit.

Setelah pergeseran bit sebanyak 16 putaran di atas, (C_i,D_i) akan mengalami permutasi kompresi menggunakan matriks PC-2 sebagaimana dalam Tabel 2.5. Dengan permutasi kompresi ini pasangan (C_i,D_i) yang berukuran 56 bit diturunkan menjadi kunci internal K_i (untuk $1 \le i \le 16$) yang berukuran 48 bit.

Dalam hal ini K_i merupakan penggabungan dari bit-bit C_i pada posisi :

14, 17, 11, 24, 1, 5, 3, 28, 15, 6, 21, 10 23, 19, 12, 4, 26, 8, 16,
7, 27, 20, 13, 2,

dengan bit-bit D_i pada posisi :

41, 52, 31, 37, 47, 55, 30, 40, 51, 45, 33, 48
44, 49, 39, 56, 34, 53, 46, 42, 50, 36, 29, 32.

Jadi, setiap kunci internal K_i mempunyai panjang 48 bit.

Tabel 2.5 Permutasi Kompresi 2

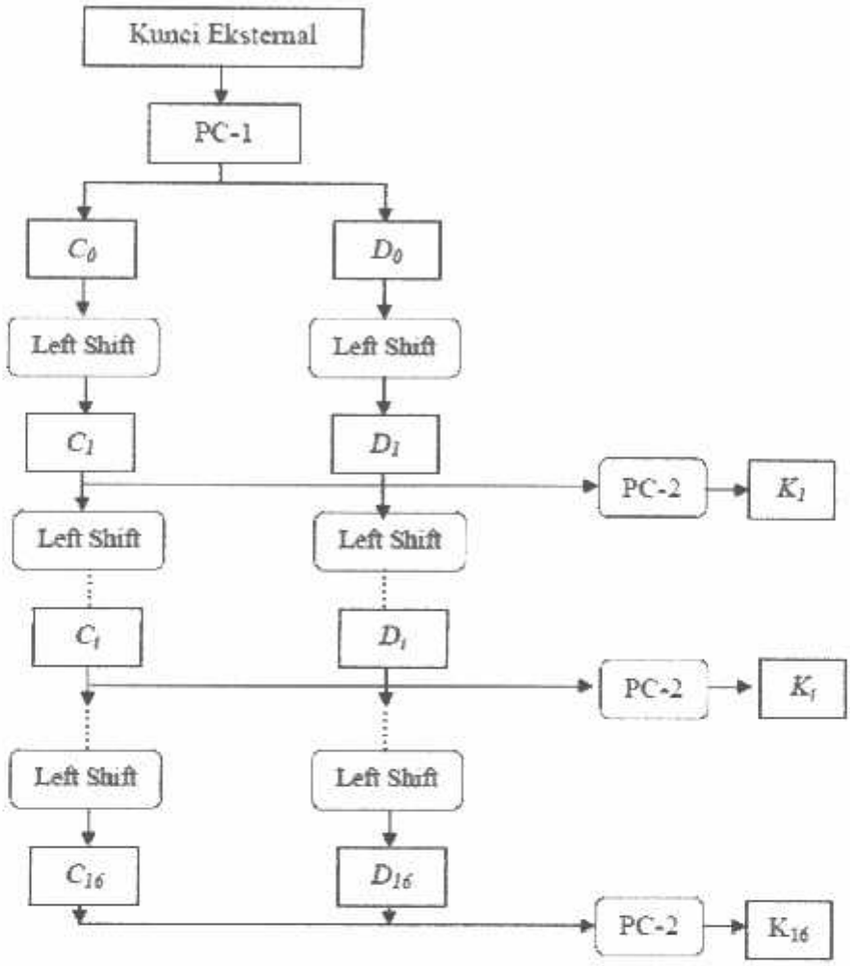
PC-2					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Input pertama dalam Tabel 2.5 adalah 14 yang artinya bahwa bit ke-14 dari (C_i,D_i) merupakan bit ke-1 dari K_i, bit ke-17 dari (C_i,D_i) merupakan bit ke-2 dari K_i, dan seterusnya sampai bit ke-32 dari (C_i,D_i) merupakan bit ke-48 dari K_i.

Bila jumlah pergeseran bit-bit pada Tabel 2.3 dijumlahkan semua, jumlah seluruhnya menjadi 28 yang sama dengan jumlah bit pada C_i dan D_i. Setelah putaran ke-16 didapatkan kembali $C_{16} = C_0$ dan $D_{16} = D_0$.

Pembangkitan kunci internal untuk proses dekripsi sama seperti pada proses enkripsi dengan pergeseran ke kiri diganti pergeseran ke kanan *(right*

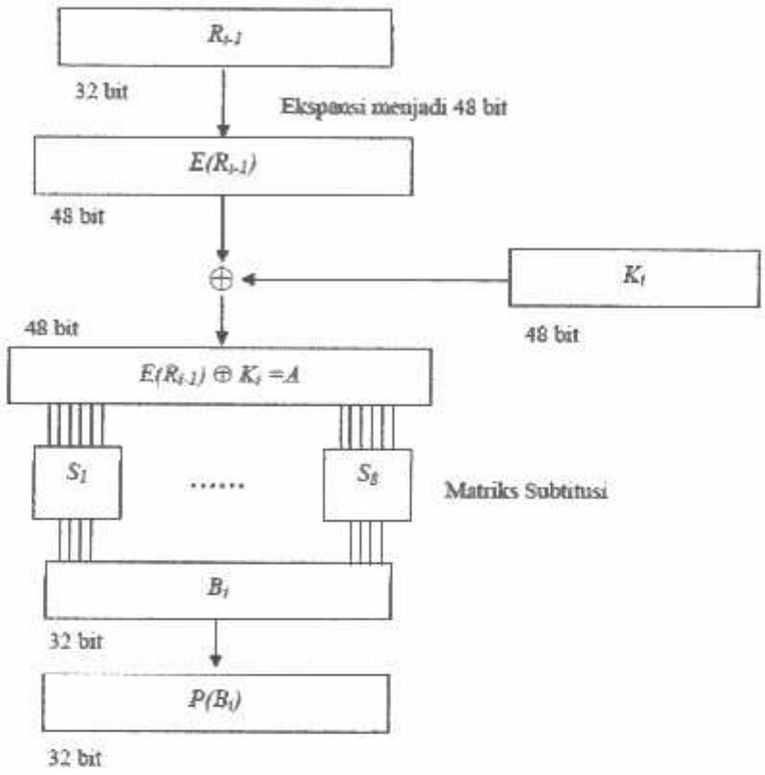
shift). Detail proses pembangkitan 16 buah kunci internal K_i dari kunci eksternal ini ditunjukkan dalam Gambar 2.5.



Gambar 2.5 Diagram alir pembangkitan kunci-kunci internal DES

2.2.5 Enkripsi Dan Dekripsi Algoritma DES

Sebagaimana dijelaskan sebelumnya, proses enkripsi terhadap blok *plaintext* yang panjangnya 64 bit dilakukan setelah permutasi awal terhadap blok *plaintext* tersebut. Setelah itu blok *plaintext* dibagi menjadi dua bagian yaitu kiri (*L*) dan kanan (*R*) yang masing-masing panjangnya 32 bit. Kemudian kedua bagian ini mengalami 16 kali putaran dengan setiap putaran menggunakan kunci internal yang berbeda. Setiap satu putaran merupakan jaringan Feistel yang secara matematis dinyatakan dalam Persamaan 2.1. Dalam persamaan tersebut fungsi *f* merupakan fungsi tranformasi yang merupakan inti dari algoritma DES, diperlihatkan dalam Gambar 2.6.



Gambar 2.6 Diagram alir fungsi f algoritma DES

Dalam gambar tersebut R_{i-1} diperoleh dari blok kanan sebuah *plaintext* kemudian menjadi input dari transformasi *f*. Fungsi ekspansi *E* berguna untuk memperluas blok R_{i-1} yang panjangnya 32 bit menjadi blok yang panjangnya 48 bit dengan matriks permutasi ekspansi *(expansion permutation)* sebagaimana dalam Tabel 2.6. Dalam matriks permutasi ekspansi ini blok input diperluas dengan melakukan penambahan sejumlah 16 bit (bilangan yang dicetak tebal) ke dalam Tabel 2.6.

Input pertama dalam Tabel 2.6 adalah 32 yang artinya bit ke-32 dari R_{i-1} merupakan bit ke-1 dari $E(R_{i-1})$, bit ke-1 dari R_{i-1} merupakan bit ke-2 dari $E(R_{i-1})$ dan seterusnya sampai bit ke-1 dari R_{i-1} merupakan bit ke-48 dari $E(R_{i-1})$.

Tabel 2.6 Permutasi 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

Selanjutnya hasil ekspansi, $E(R_{i-1})$, yang panjangnya 48 bit di-XOR-kan dengan kunci internal K_i yang panjangnya 48 bit menghasilkan vektor A yang panjangnya 48 bit.

$$E(R_{i-1}) \oplus K_i = A \qquad (2.2)$$

Vektor A tersebut dikelompokkan menjadi delapan kelompok yang masing-masing kelompok terdiri dari 6 bit. Misalkan $A=A_1A_2A_3A_4A_5A_6A_7A_8$ dengan A_1 sampai dengan A_8 merupakan kelompok dari A. Delapan kelompok tersebut akan menjadi input untuk proses subtitusi yang dilakukan menggunakan delapan buah S-Box yaitu S_1, S_2, ...S_8. Setiap S-Box menerima input sebanyak 6 bit dan menghasilkan output sebanyak 4 bit. Kelompok 6 bit pertama menggunakan S_1, kelompok 6 bit kedua menggunakan S_2, dan seterusnya sampai kelompok 6 bit kedelapan menggunakan S_8.

Misalkan delapan kelompok A dinyatakan dengan A_j ($1 \le j \le 8$), $A_j=a_1a_2a_3a_4a_5a_6$ dengan a_1 sampai dengan a_6 merupakan 6 bit dalam A_j. Bit pertama dan terakhir dalam A_j (a_1 dan a_6) digunakan untuk menunjukkan indek baris ke-r dari S_j (S-Box ke-j) dengan $0 \le r \le 3$. Sedangkan empat bit yang tengah ($a_2a_3a_4a_5$) digunakan untuk menunjukkan indek kolom ke-c dari S_j dengan $0 \le c \le 15$. Output dari subtitusi S-Box ini adalah vektor *B* yang panjangnya 32 bit. Vektor B terbagi menjadi delapan kelompok yang masing-masing kelompok terdiri dari empat bit. Misalkan $B=B_1B_2B_3B_4B_5B_6B_7B_8$ dengan B_1 sampai dengan B_8 merupakan kelompok dari B. Secara umum kelompok dari B tersebut dapat dinyatakan dengan B_j. Untuk memperoleh B_j dilakukan dengan menggunakan A_j

sebagai input dalam proses S-Box. Secara umum proses mendapatkan output dari S-Box dinyatakan dalam Persamaan 2.3.

$$B_j = S_j (A_j) = S_j (r,c) \text{ untuk } 1 \leq j \leq 8 \quad (2.3)$$

Selanjutnya vektor *B* ini menjadi input untuk proses permutasi menggunakan matriks permutasi P (P-Box) seperti Tabel 2.7 sehingga dihasilkan *P(B)*. Tujuan permutasi P ini adalah untuk mengacak hasil proses subtitusi S-Box.

Tabel 2.7 P *Permutation*

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

Input pertama dalam Tabel 2.7 adalah 16 yang artinya bit ke-16 dari *B* merupakan bit ke-1 dari *P(B)*, bit ke-7 dari *B* merupakan bit ke-2 dari *P(B)*, dan seterusnya sampai bit ke-25 dari *B* merupakan bit ke-32 dari *P(B)*.
P(B) yang panjangnya 32 bit ini merupakan output dari fungsi *f*.

$$P(B) = f(R_{i-1}, K_i) \quad (2.4)$$

Selanjutnya *P(B)* di-XOR-kan dengan L_{i-1} sehingga diperoleh R_i.

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

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

Jadi output dari putaran ke-*i* untuk 1£*i*£16 adalah,

$$(L_i, R_i) = (R_{i-1}, L_{i-1} \oplus f(R_{i-1}, K_i)) \quad (2.5)$$

Dari Gambar 2.4 dapat diketahui jika (L_{16}, R_{16}) merupakan hasil dari putaran ke-16, maka (R_{16}, L_{16}) merupakan pra-*ciphertext* dari proses enkripsi algoritma DES . *Chipertext* yang sebenarnya dapat diperoleh dengan melakukan *invers* permutasi awal (IP^{-1}) terhadap blok pra-*ciphertext* tersebut.

Sedangkan proses dekripsi DES dimulai dengan putaran ke-16, 15,...,2, 1. Untuk setiap putaran dalam proses dekripsi, dihasilkan output yang secara matematis dapat dinyatakan dalam Persamaan 2.6. Persamaan ini diturunkan dari Persamaan 2.1 yang merupakan output dari setiap putaran dalam proses enkripsi.

$Ri\text{-}1 = Li$

$$L_{i-1} = R_i \oplus f(L_i, K_i) \quad (2.6)$$

Dalam hal ini (R_{16}, L_{16}) adalah input awal untuk proses dekripsi. Untuk memperoleh blok (R_{16}, L_{16}) maka dilakukan permutasi terhadap *ciphertext* menggunakan matriks permutasi IP^{-1} seperti dalam Tabel 2.8. Output terakhir dari proses dekripsi adalah (L_0, R_0). Selanjutnya dilakukan permutasi terhadap (L_0, R_0) menggunakan matriks permutasi awal pada Tabel 2.5 sehingga diperoleh kembali blok *plaintext* semula.

Proses pembangkitan kunci internal dalam proses dekripsi pada prinsipnya hampir sama dengan proses enkripsi sebagaimana dalam Gambar 2.5. Gambar tersebut menunjukkan K_{16} dihasilkan dari (C_{16}, D_{16}) setelah mengalami permutasi kompresi menggunakan matriks PC-2 dalam Tabel 2.5 pada permutasi akhir. Sebagaimana telah dijelaskan di depan bahwa setelah putaran ke-16 dari proses pembangkitan kunci internal, didapatkan kembali $C_{16} = C_0$ dan $D_{16} = D_0$, sehingga K_{16} dapat dihasilkan dari (C_0, D_0) tanpa harus melakukan pergeseran bit lagi. Perlu diingat (C_0, D_0) merupakan bit-bit dari kunci eksternal K yang diberikan oleh pengguna pada waktu proses dekripsi.

Selanjutnya K_{15} diperoleh dari (C_{15}, D_{15}). Untuk mendapatkan (C_{15}, D_{15}) diperoleh dengan cara menggeser C_{16} dan D_{16} satu bit ke kanan. Untuk K_{14} sampai K_1 diperoleh dari (C_{14}, D_{14}) sampai (C_1, D_1). Secara umum dituliskan untuk mendapatkan (C_{i-1}, D_{i-1}) diperoleh dengan cara menggeser C_i dan D_i

sepanjang satu atau dua bit menggunakan Tabel 2.4, tetapi pergeseran ke kiri diganti menjadi pergeseran ke kanan.

2.2.6 Permutasi Terakhir

Permutasi terakhir ini dilakukan setelah 16 kali putaran terhadap gabungan blok kiri dan blok kanan. Hasil setelah dilakukan 16 kali putaran tersebut adalah (L_{16}, R_{16}). Kemudian hasil dari putaran ini di permutasi menggunakan matriks permutasi awal balikan seperti dalam Tabel 2.8.

Tabel 2.8 *Invers Initial Permutation*

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

Input pertama dalam Tabel 2.8 adalah 40 yang artinya bahwa bit ke-40 dari (L_{16}, R_{16}) merupakan bit ke-1 dari IP^{-1}, bit ke-8 dari (L_{16}, R_{16}) merupakan bit ke-2 dari IP^{-1} dan seterusnya sampai bit ke-25 dari (L_{16}, R_{16}) merupakan bit ke-64 dari IP^{-1}.

Perlu diketahui, tabel permutasi *IP, Invers IP, S-Box, P, E, PC-1 dan PC-2* pada algoritma DES isinya merupakan ketetapan dari algoritma DES

2.3 Blowfish

Blowfish adalah sebuah algoritma kriptografi yang beroperasi pada mode blok (Agustinus, 2009:2). Aryus Doni (2008:204) mengemukana bahwa Browfish *merupakan sebuah algoritma kunci simetri blok kode dirancang pada tahun 1993*

oleh Bruce Schneier untuk mengganti DES. Pada saat itu banyak sekali rancangan algoritma yang ditawarkan, namun hamper semua terhalang oleh paten atau kerahasian pemerintah Amerika, Schneier menyatakan Browfish bebas paten dan akan diletakkan pada domain public.

Blowfish alias "*OpenPGP.Cipher.4*" merupakan enkripsi yang termasuk dalam golongan *Symmetric Cryptosystem* metoda enkripsinya mirip dengan DES (DES-like Cipher) 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 (32bit keatas dengan *cache* data yang besar). Agustinus (2009:5) juga berpendapat bahwa Browfish adalah alogaritma kunci simetri, yang berarti menggunakan kunci yang sama untuk melakukan enkripsi dan dekripsi berkas. Browfish juga merupakan cipher blok, yang berarti selama proses enkripsi dan dekripsi, Blowfish akan membagi pesan menjadi blok-blok dengan ukuran yang sama panjang.

Aryus Dony (2008:204) menyatakan bahwa, *Blowfish* dikembangkan untuk memenuhi kriteria desain yang cepat dalam implementasinya dimana pada keadaan optimal dapat mencapai 26 *clock cycle per byte*, kompak dimana dapat berjalan pada memori kurang dari 5 KB, sederhana dalam algoritmanya sehingga mudah diketahui kesalahannya, dan keamanan yang variabel dimana panjang kunci bervariasi (minimum 32 bit, maksimum 448 bit, *Multiple* 8 bit, *default* 128 bit). Stiawan Deris (2005:75) berpendapat bahwa, Blowfish dioptimasikan untuk berbagai aplikasi dimana kunci tidak sering berubah, seperti pada jaringan komunikasi atau enkripsi file secara otomatis. Dalam pengimplementasiannya dalam komputer bermicroprosesor 32-bit dengan *cache* data yang besar (Pentium dan Power PC) Blowfish terbukti jauh lebih cepat dari DES. Tetapi Blowfish tidak cocok dengan aplikasi dengan perubahan kunci yang sering atau sebagai fungsi hast satu arah seperti pada aplikasi *packet switching*. Blowfish pun tidak dapat digunakan pada aplikasi kartu pintar (*smart card*) karena memerlukan memori yang besar.

Menurut Aryus Doni (2008:207) *Blowfish* termasuk dalam enkripsi block Cipher 64-bit dangan panjang kunci yang bervariasi antara 32-bit sampai 448-bit Algoritma Blowfish terdiri atas dua bagian:

1. **Key-Expansion**

   Berfungsi merubah kunci (Minimum 32-bit, Maksimum 448-bit) menjadi beberapa array subkunci (subkey) dengan total 4168 byte.

2. **Enkripsi Data**

   Terdiri dari iterasi fungsi sederhana (*Feistel Network*) sebanyak 16 kali putaran. Setiap putaran terdiri dari permutasi kunci-*dependent* dan substitusi kunci- dan data-*dependent*. Semua operasi adalah penambahan (*addition*) dan XOR pada variabel 32-bit. Operasi tambahan lainnya hanyalah empat penelusuran tabel (*table lookup*) array berindeks untuk setiap putaran.

2.3.1 Enkripsi Dan Dekripsi Algoritma Blowfish

Blowfish menggunakan subkunci yang besar. Kunci tersebut harus dihitung sebelum enkripsi atau dekripsi data. *Blowfish* adalah algoritma yang menerapkan jaringan Feistel (*Feistel Network*) yang terdiri dari 16 putaran. Input adalah elemen 64-bit, X. Stiawan Deris (2005:76) berpendapat bahwa, untuk alur algoritma enkripsi dengan metoda Blowfish dijelaskan sebagai berikut:

1. Bentuk inisial P-array sebanyak 18 buah (P1,P2,.....P18) masing-masing bernilai 32-bit. Array P terdiri dari delapan belas kunci 32-bit subkunci:

   $P_1, P_2, \ldots\ldots, P_{18}$

2. Bentuk S-box sebanyak 4 buah masing-masing bernilai 32-bit yang memiliki masukan 256. Empat 32-bit S-box masing-masing mempunyai 256 entri :

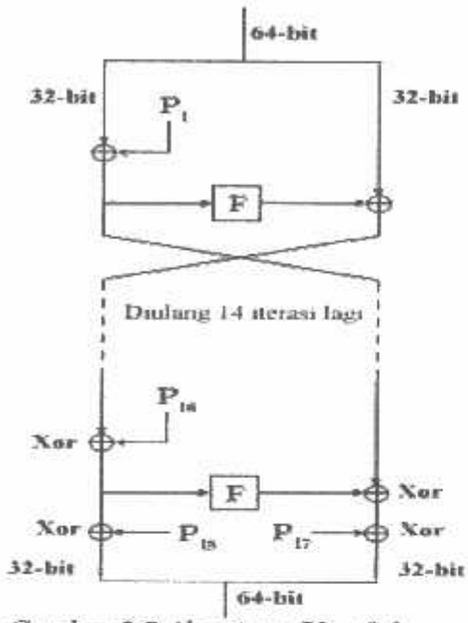
   $S_{1,0}, S_{1,1}, \ldots\ldots\ldots\ldots, S_{1,255}$

$S_{2,0}, S_{2,1}, \ldots\ldots\ldots\ldots, S_{2,255}$

$S_{3,0}, S_{3,1}, \ldots\ldots\ldots\ldots, S_{3,255}$

$S_{4,0}, S_{4,1}, \ldots\ldots\ldots\ldots, S_{4,255}$

3. Plaintext yang akan dienkripsi diasumsikan sebagai masukan, Plaintext tersebut diambil sebanyak 64-bit, dan apabila kurang dari 64-bit maka kita tambahkan bitnya, supaya dalam operasi nanti sesuai dengan datanya.
4. Hasil pengambilan tadi dibagi 2, 32-bit pertama disebut XL, 32-bit yang kedua disebut XR.
5. Selanjutnya lakukan operasi XL = XL xor Pi dan XR = F(XL) xor XR
6. Hasil dari operasi diatas ditukar XL menjadi XR dan XR menjadi XL.
7. Lakukan sebanyak 16 kali, perulangan yang ke-16 lakukan lagi proses penukaran XL dan XR.
8. Pada proses ke-17 lakukan operasi untuk XR = XR xor P17 dan XL = XL xor P18.
9. Proses terakhir satukan kembali XL dan XR sehingga menjadi 64-bit kembali.



Gambar 2.7 Algoritma Blowfish

Fungsi F adalah sebagai berikut:

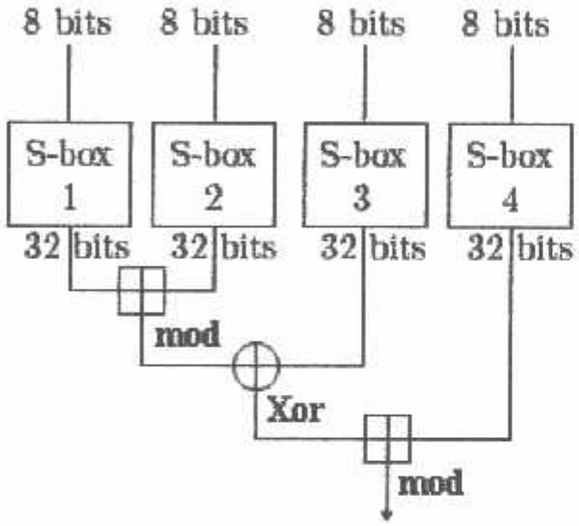
Bagi XL, menjadi empat bagian 8-bit: a,b,c dan d.

$$F(XL) = ((S1,a + S2,b \bmod 2^{32}) \text{xor } S3,c) + S4,c \bmod 2^{32} \quad (2.7)$$

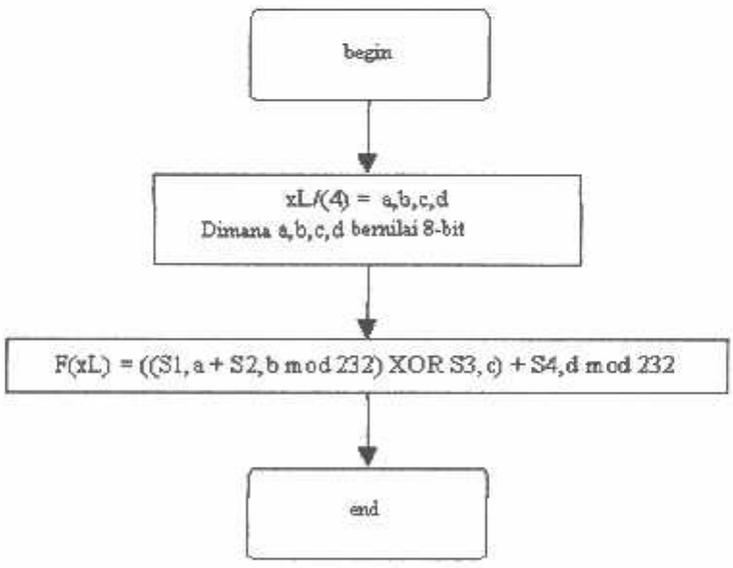
Subkunci dihitung menggunakan algoritma Blowfish, metodanya adalah sebagai berikut:

1. Pertama-tama inilialisasi P-array dan kemudian empat S-box secara berurutan dengan string yang tetap. String ini terdiri atas digit hexadesimal dari Pi.
2. XOR P1 dengan 32-bit pertama kunci, XOR P2 dengan 32-bit kedua dari kunci dan seterusnya untuk setiap bit dari kunci (sampai P18). Ulangi terhadap bit kunci sampai seluruh P-array di XOR dengan bit kunci.
3. Enkrip semua string nol dengan algoritma Blowfish dengan menggunakan subkunci seperti dijelaskan pada langkah (1) dan (2).
4. Ganti P1 dan P2 dengan keluaran dari langkah (3).

5. Enkrip keluaran dari langkah (3) dengan algoritma Blowfish dengan subkunci yang sudah dimodifikasi.
6. Ganti P3 dan P4 dengan keluaran dari langkah (5).
7. Lanjutkan proses tersebut, ganti seluruh elemen dari P-array, kemudian seluruh keempat S-box berurutan, dengan keluaran yang berubah secara kontiyu dari algoritma Blowfish.



Gambar 2.8 fungsi F dalam Blowfish



Gambar 2.9 FlowChart F fungsi

Total yang diperlukan adalah 521 iterasi untuk menghasilkan semua subkunci yang dibutuhkan. Aplikasi kemudian dapat menyimpan subkunci ini dan tidak membutuhkan langkah-langkah proses penurunan berulang kali, kecuali kunci yang digunakan berubah.

Untuk deskripsi sama persis dengan enkripsi, kecuali pada P-array (P1,P2,.....,P18) digunakan dengan urutan terbalik atau di inverskan.

2.4 Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, middleware dan aplikasi. Menurut *Rifki* (2016:5), Android adalah sebuah sistem operasi untuk perangkat mobile berbasis linux yang mencakup sistem operasi, middleware dan aplikasi. Android dibuatmenggunakan kernel linux yang dimodifikasi. Android menyediakan platformterbuka bagi para pengembang untuk menciptakan aplikasi mereka sendirisehingga dapat digunakan olch bermacam peranti penggerak

Safaat (2012) berpendapat bahwa, Android adalah sebuah kumpulan perangkat lunak untuk perangkat mobole yang mencakup system operasi, middleware dan aplikasi utama mobile. Menurut Safaat (2012) Android memiliki 4 (empat) karakteristik sebagai berikut:

1. Terbuka

   Android dibangun untuk benar-benar terbuka sehingga sebuah aplikasi dapat memanggil salah satu fungsi inti ponsel seperti membuat panggilan, mengirim pesan teks, menggunakan kamera, dan lain-lain. Android menggunakan sebuah mesin virtual yang dirancang khusus untuk mengoptimalkan sumber daya memeri dan perangkat keras yang terdapat di dalam perangkat.
2. Semua Aplikasi dibuat sama

   Android tidak memberikan perbedaan terhadap aplikasi utama dari telepon dan aplikasi pihak ketiga (third-party application). Semua aplikasi dapat dibangun utuk memliki akses yang sama terhadap

kemmapuan sebuah telepon dalam menyiadakan layanan dan aplikasi yang luas terhadap para pengguna.

3. Memecahkan hambatan pada aplikasi

   Android memecah hambatan utuk membangun aplikasi yang baru dan inovatif. Misalnya, pengembang dapat menggabungkan informasi yang diperoleh dari web dengan data pada ponsel seseorang seperti kontak pengguna, kalender, atau lokasi geografis.

4. Pengembangan aplikasi yang cepat dan mudah

   Android menyediakan akses yang sangat luas kepada pengguna untuk menggunakan library yang dipergunakan tools yang dapat digunakan untuk membangun aplikasi yang semakin baik. Android memiliki sekumpulan tools yang dapat digunakan sehingga membantu para pengembang dalam meningkatkan produktivitas pada saat membangun aplikasi yang dibuat.

Android merupakan system operasi berbasis Linux yang bersifat terbuka (open source) dan dirancang untuk perangkat seluler layer sentuh seperti smartphone dan computer tablet. Android dikembangkan oleh Android, Inc., dengan dukungan financial dari google yang kemudian dibeli pada tahun 2005.Tampilan Android didasarkan pada manipulasi langsung, menggunakan masukan sentuh yang serupa dengan tindakan di dunia nyata, seperti menggesek, mengetuk, mencubut, dan membalikkan cubitan untuk memanipulasi obyek di layar (Salbino Sherief , 2014:7).

Para penggemar open source kemudian membangun komunitas yang membangun dan berbagi *Android berbasis firmware* dengan sejumlah penyesuaian dan fitur-fitur tambahan, seperti *FLAC lossless audio* dan kemampuan untuk menyimpan download *aplikasi pada microSD card.* Mereka sering memperbaharui paket-pakct firmware dan menggabungkan elemen-elemen fungsi Android yang belum resmi diluncurkan dalam suatu *carrier-sanction firmware* (Safaat, 2012:12).

Berdasarkan uraian diatas dapat disimpulakan bahwa Android merupakan sistem operasi yang berbasis Linux untuk telepon seluler seperti telepon pintar dan komputer tablet. Android menyediakan platform terbuka bagi para

pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak.

2.5 SMS (*Short Message Service*)

Short Message Service atau lebih dikenal dengan SMS saat ini sudah banyak digunakan oleh masyarakat sebagai alat komunikasi. Kelebihan dari SMS adalah biayanya yang murah, cepat, dan langsung pada tujuan. Mulai dari anak – anak, remaja, orang dewasa hingga orang lanjut usia hampir semuanya memanfaatkan teknologi SMS ini. Awalnya SMS hanya digunakan untuk berkomunikasi antar personal saja. Tetapi kini seiring perkembangan jaman, penggunaan SMS semakin berkembang. Misalnya untuk *polling* suatu audisi, mengakses nilai kuliah, mengirim kritik dan saran kepada media massa untuk kemudian dipublikasikan dan banyak lagi penggunaan lainnya (Sasongko, 2008 dalam Zahra Khadijah, 2013:4).

SMS (*Short Message Service*) adalah merupakan salah satu layanan pesan teks yang dikembangkan dan distandardisasi oleh suatu badan bernama ETSI (*European Telecommunication Standards Institute*) sebagian dari pengembangan GSM *(Global System for Mobile Communication) Phase* 2, yang terdapat pada dokumentasi GSM 03.40 dan GSM 03.38. Fitur SMS ini memungkinkan perangkat Stasiun Seluler Digital (*Digital Cellular Terminal*, seperti Ponsel) untuk dapat mengirim dan menerima pesan-pesan teks dengan panjang sampai dengan 160 karakter melalui jaringan GSM (Wahana Komputer, 2005:12).

SMS berupa pesan teks, jumlah karakter pada setiap pengiriman bergantung pada operatornya. Operator selular di Indonesia umumnya membatasi 160 karakter untuk satu pengiriman dan penerima SMS. Selain itu SMS merupakan metode store dan forward schingga keuntungan yang didapat adalah pada saat telepon selular penerima tidak dapat dijangkau, dalam arti tidak aktif atau diluar *service area*, penerima tetap dapat menerima SMS-nya apabila telepon selular tersebut suadah aktif kembali.

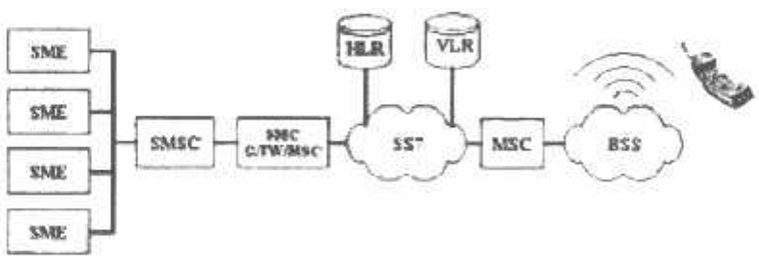
SMS atau layanan pesan singkat merupakan revolusi dalam layanan telekomunikasi, dimana layanan tidak berbasis suara melainkan layanan berupa pengiriman pesan teks singkat antar perangkat telepon selular. Layanan ini

merupakan layanan tambahan terhadap dua layanan utama (voice dan switched data) dalam sistem jaringan komunikasi GSM. Namun, karena keberhasilan SMS yang tidak terduga menjadikannya sebagai bagian integral dari layanan sistem standar-standar komunikasi lainnya, seperti CDMA, UMTS bahkan jaringan telepon rumah juga mulai mengadopsi teknologi ini.

Berdasarkan uraian diatas dapat disimpulkan bahwa, SMS (*Short Message Service*) yaitu suatu fasilitas untuk mengirim suatu pesan dan menerima singkat berupa TEKS melalui perangkat nirkabel, yaitu perangkat KOMUNIKASI teleon Selular, Dalam, Hal inisial perangkat nirkabel Yang digunakan adalah Telepon Selular. *Short Message Service* (SMS) merupakan layanan yang banyak diaplikasikan pada sistem komunikasi tanpa kabel (nirkabel), memungkinkan dilakukannya pengiriman pesan dalam bentuk alphanumeric antar terminal pelanggan atau antar terminal pelanggan dengan sistem eksternal seperti e-mail, paging, voice mail dan lain-lain.

2.5.1 Cara kerja SMS

Mekanisme cara kerja sistem SMS adalah melakukan pengiriman short message dari satu terminal pelanggan ke terminal yang lain. Hal ini dapat dilakukan berkat adanya sebuah entitas dalam sistem SMS yang bernama *Short Message Service Centre (SMSC)*, disebut juga *Message Centre (MC)*. SMSC merupakan sebuah perangkat yang melakukan tugas *store and forward trafik short message*. Didalamnya termasuk penentuan atau pencarian rute tujuan akhir dari *sort message* (Friedhelm,2010:14).



Gambar 2.10 Arsitektur dasar jaringan SMS

SMSC memiliki interkonektivitas dengan *SME (Short Messeging Entity)* yang dapat berupa jaringan e-mail, web, dan voice e-mail. SMSC inilah yang akan melakukan manajemen pesan SMS, baik untuk pengiriman, pengaturan antrian SMS, ataupun penerimaan SMS.

Sarram, Ghasemzadeh, dan Aghaei (2008) dalam Zahra Khadijah, (2013:5) menyatakan, keuntungan utama dari SMS adalah kenyamanan dan efisiensi akses dalam pemantauan dan kontrol. Para penelitian Yazd University tersebut mengambil keuntungan dari layanan pesan singkat untuk membuat sistem administrasi dan *remote control* yang terintegrasi dengan protokol SNMP. Sistem tersebut mampu menghasilkan laporan yang sesuai pada status jaringan yang kemudian ditransmisikan ke administrator jaringan dengan format SMS. Sistem administrasi tersebut juga mampu menerapkan perintah yang diperlukan secara jarak jauh melalui telepon seluler menggunakan SMS untuk mengontrol dan diambil tindakan.

Supriyono dan Jatmiko (2008: 34) dalam Zahra Khadijah, (2013:7) menyatakan bahwa pemanfaatan layanan SMS juga diterapkan pada pengembangan tulisan berjalan (*running text*) pada *dot matriks*. Pengembangan sistem tersebut membuktikan bahwa komunikasi serial antara mikrokontroler dan *handphone* berhasil dilakukan dan diuji menggunakan fasilitas *hyperterminal* pada PC.

Lim Sandjaja, dan Yulimingtarto (2005) dalam Zahra Khadijah, (2013:9) menyatakan bahwa penggunaan teknologi SMS sebagai media komunikasi dalam sistem mikroprosesor yang dilengkapi dengan GPS dan telepon seluler yang ditempatkan pada kendaraan memiliki keterbatasan yaitu tergantung kepada operator yang digunakan. Jika jaringan operator tidak sibuk, maka pengiriman SMS akan cepat. Tetapi jika operator sedang sibuk, maka pengiriman SMS menjadi lebih lambat bahkan bisa gagal.

Layanan SMS merupakan sebuah layanan yang bersifat non-real time dimana sebuah short message dapat disubmit ke suatu tujuan, tidak peduli apakah tujuan tersebut aktif atau tidak. Bila dideteksi tujuan tidak aktif, maka sistem akan menunda pengiriman ke tujuan hingga tujuan aktif kembali. Pada dasarnya

BAB III

ANALISIS DAN PERANCANGAN

3.1 Analisis

3.1.1 Sistem Saat Ini

Layanan SMS *(Short Message Service)* sebagai layanan pertukaran informasi atau pesan pendek menjadi 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. Celah keamanan terbesar pada layanan komunikasi SMS adalah pada saat SMS tersebut sedang dikirim melalui jaringan SMS tersebut. SMS bekerja pada jaringan nirkabel yang memungkinkan terjadinya pencurian isi pesan SMS ketika dalam proses transmisi dari pengirim ke penerima, kasus ini disebut SMS *interception.*

3.1.2 Contoh Kasus Dan Permasalahan

Untuk memperjelas bagaimana proses enkripsi dan dekripsi menggunakan algoritma DES, berikut diberikan satu contoh kasus implementasi algoritma DES dalam proses enkripsi dan dekripsi.

Ahmad bermaksud mengirimkan sebuah pesan melalui jaringan tertentu kepada Ibrahim. Karena Ahmad menginginkan hanya Ibrahim yang tahu pesannya, dia melakukan enkripsi pada pesan *plaintext*-nya menggunakan algoritma DES sehingga dihasilkan *ciphertext*. Ahmad berharap hanya Ibrahim yang bisa melakukan dekripsi terhadap *ciphertext* yang dikirimnya. Pesan Ahmad berupa *plaintext* P = 0123456789ABCDEF dan kunci eksternal yang digunakan untuk melakukan enkripsi adalah K = 133457799BBCDFF1. Proses enkripsi menggunakan algoritma DES terhadap *plaintext* Ahmad dilakukan dengan langkah-langkah sebagai berikut:

1. Permutasi Awal

Sebelum dilakukan permutasi awal terlebih dahulu *plaintext* P yang masih berupa bilangan hexadesimal dikonversi ke dalam bilangan biner menggunakan Tabel 2.1. Hasil konversi berupa bilangan biner dengan panjang 64 bit sebagai berikut :

P = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111.

Selanjutnya dilakukan permutasi awal terhadap blok *plaintext* tersebut menggunakan matriks permutasi awal dalam Tabel 2.2 sehingga diperoleh,

IP = 1100 1100 0000 0000 1100 1100 1111 1111 1111 0000 1010 1010 1111 0000 1010 1010.

2. Menentukan Kunci Internal

Untuk melakukan sebuah proses enkripsi menggunakan algoritma DES dibutuhkan kunci internal sebanyak 16 buah yang dibangkitkan dari kunci eksternal. Kunci eksternal K = 133457799BBCDFF1 berupa bilangan hexadesimal terlebih dahulu dikonversi ke dalam bilangan biner menggunakan Tabel 2.1. Kunci eksternal yang tersusun dari 64 bit, yaitu

K = 00010011 00110100 01010111 01111001 10011011 10111100 1101111 11110001.

Kunci eksternal K ini menjadi input untuk permutasi menggunakan permutasi kompresi Tabel 2.3 sehingga dihasilkan $K+$ sepanjang 56 bit, yaitu

$K+$ = 1111000 0110011 0010101 0101111 0101010 1011001 1001111 0001111.

Selanjutnya 56 bit dari $K+$ ini dibagi menjadi dua blok yaitu blok kiri (C_0) sepanjang 28 bit dan blok kanan (D_0) sepanjang 28 bit :

C_0 = 1111000 0110011 0010101 0101111

D_0 = 0101010 1011001 1001111 0001111.

Kedua bagian ini digeser ke kiri sepanjang satu atau dua putaran dengan jumlah pergeseran pada setiap putaran menggunakan Tabel 2.4, diperoleh C_i dan D_i untuk $1 \leq i \leq 16$ sebagai berikut :

C_{16} = 1111000 0110011 0010101 0101111

D_{16} = 0101010 1011001 1001111 0001111.

Selanjutnya (C_i, D_i) hasil pergeseran tersebut mengalami permutasi kompresi menggunakan matriks PC-2 dalam Tabel 2.5. Dengan permutasi kompresi ini pasangan (C_i,D_i) yang berukuran 56 bit diturunkan menjadi kunci internal K_i (untuk $1 \le i \le 16$) yang berukuran 48 bit.

K_1 = 000110 110000 001011 101111 111111 000111 000001 110010

K_2 = 011110 011010 111011 011001 110110 111100 100111 100101

K_3 = 010101 011111 110010 001010 010000 101100 111110 011001

K_4 = 011100 101010 110111 010110 110110 110011 010100 011101

K_5 = 011111 001110 110000 000111 111010 110101 001110 101000

K_6 = 011000 111010 010100 111110 010100 000111 101100 101111

K_7 = 111011 001000 010010 110111 111101 100001 100010 111100

K_8 = 111101 111000 101000 111010 110000 010011 101111 111011

K_9 = 111000 001101 101111 101011 111011 011110 011110 000001

K_{10} = 101100 011111 001101 000111 101110 100100 011001 001111

K_{11} = 001000 010101 111111 010011 110111 101101 001110 000110

K_{12} = 011101 010111 000111 110101 100101 000110 011111 101001

K_{13} = 100101 111100 010111 010001 111110 101011 101001 000001

K_{14} = 010111 110100 001110 110111 111100 101110 011100 111010

K_{15} = 101111 111001 000110 001101 001111 010011 111100 001010

K_{16} = 110010 110011 110110 001011 000011 100001 011111 110101.

3. Proses Enkripsi DES

Di dalam proses enkripsi, setelah blok *plaintext* dikenakan permutasi awal diperoleh IP. Selanjutnya IP dibagi menjadi dua bagian yaitu blok kiri (L_0) dan blok kanan (R_0) yang masing-masing panjangnya 32 bit.

L_0 = 1100 1100 0000 0000 1100 1100 1111 1111

R_0 = 1111 0000 1010 1010 1111 0000 1010 1010.

Selanjutnya kedua blok ini diproses kedalam 16 putaran DES dengan satu putaran DES merupakan model jaringan Feistel dan secara matematis dinyatakan dalam Persamaan 2.1.

Menggunakan Persamaan 2.1 untuk $i = 1$ diperoleh,

K_1 = 000110 110000 001011 101111 111111 000111 000001 110010

$L_1 = R_0$ = 1111 0000 1010 1010 1111 0000 1010 1010

$R_1 = L_0 \oplus f(R_0, K_1)$.

Untuk mendapatkan nilai dari R_1, terlebih dahulu dicari nilai dari $f(R_0, K_1)$ menggunakan diagram alur fungsi f. Dalam hal ini R_0 merupakan input dari tranformasi f. Selanjutnya menggunakan fungsi ekspansi, R_0 yang panjangnya 32 bit diperluas menjadi blok yang panjangnya 48 bit dengan matriks permutasi ekspansi dalam Tabel 2.6, diperoleh $E(R_0)$ sebagai berikut,

R_0 = 1111 0000 1010 1010 1111 0000 1010 1010

$E(R_0)$ = 011110 100001 010101 010101 011110 100001 010101 010101.

Selanjutnya hasil ekspansi, $E(R_0)$, di-XOR-kan dengan kunci internal K_1 menghasilkan vektor A yang panjangnya 48 bit. Dengan Persamaan 2.2 diperoleh vektor A sebagai berikut,

K_1 = 000110 110000 001011 101111 111111 000111 000001 110010

$E(R_0)$ = 011110 100001 010101 010101 011110 100001 010101 010101

$E(R_0) \oplus K_1 = A$ = 011000 010001 011110 111010 100001 100110 010100 100111.

Vektor A yang telah dikelompokkan menjadi delapan kelompok dan masing-masing kelompok terdiri dari enam bit menjadi input untuk proses subtitusi S-Box.

A_1= 011000, A_2=010001,....., A_7= 010100, A_8=100111.

Dengan Persamaan 2.3 untuk $1 \leq j \leq 8$ diperoleh,

$B_1 = S_1\ (A_1) = S_1\ (r, c) = S_1(00, 1100) = S_1\ (0, 12) = 5 = 0101$

$B_2 = S_2\ (A_2) = S_2\ (r, c) = S_2\ (01, 1000) = S_2\ (1, 8) = 12 = 1100$

$B_3 = S_3\ (A_3) = S_3\ (r, c) = S_3\ (00, 1111) = S_3\ (0, 15) = 8 = 1000$

$B_4 = S_4\ (A_4) = S_4\ (r, c) = S_4\ (10, 1101) = S_4\ (2, 13) = 2 = 0010$

$B_5 = S_5\ (A_5) = S_5\ (r, c) = S_5\ (11, 0000) = S_5\ (3, 0) = 11 = 1011$

$B_6 = S_6\ (A_6) = S_6\ (r, c) = S_6\ (10, 0011) = S_6\ (2, 3) = 5 = 0101$

$B_7 = S_7\ (A_7) = S_7\ (r, c) = S_7\ (00, 1010) = S_7\ (0, 10) = 9 = 1001$

$B_8 = S_8\ (A_8) = S_8\ (r, c) = S_8\ (11, 0011) = S_8\ (3, 3) = 7 = 0111.$

Sehingga diperoleh output dari proses S-Box sebagai berikut,

$B = B_1B_2B_3B_4B_5B_6B_7B_8$ = 0101 1100 1000 0010 1011 0101 1001 0111.

Selanjutnya vektor *B* ini menjadi input untuk proses permutasi menggunakan matriks permutasi seperti Tabel 2.7 sehingga dihasilkan *P(B)*, yaitu *P(B)* = 0010 0011 0100 1010 1010 1001 1011 1011.

P(B) merupakan output dari fungsi *f*. Dengan Persamaan 2.4 untuk $i = 1$, diperoleh $P(B) = f(R_0, K_1)$. Selanjutnya untuk mendapatkan R_1 dilakukan dengan meng-XOR-kan $f(R_0, K_1)$ dengan L_0 sehingga diperoleh R_1 sebagai berikut,

L_0 – 1100 1100 0000 0000 1100 1100 1111 1111

$f(R_0, K_1)$ = 0010 0011 0100 1010 1010 1001 1011 1011

$L0 \oplus f(R_0, K_1)$ = 1110 1111 0100 1010 0110 0101 0100 0100.

Jadi, $R_1 = L_0 \oplus f(R_0, K_1)$ = 1110 1111 0100 1010 0110 0101 0100 0100.

Proses di atas (untuk i=1) merupakan putaran pertama proses enkripsi menggunakan algoritma DES. Selanjutnya dengan cara yang sama dilakukan putaran ke-2 (i=2) dan seterusnya sampai dengan putaran ke-16 (i=16).

Untuk putaran ke-16 dari algoritma DES diperoleh data sebagai berikut,

K_{16} = 110010 110011 110110 001011 000011 100001 011111 110101

$L_{16} = R_{15}$ = 0100 0011 0100 0010 0011 0010 0011 0100

$E(R_{15})$ = 001000 000110 101000 000100 000110 100100 000110 101000

$E(R_{15}) \oplus K_{16} = A$ = 111010 110101 011110 001111 000101 000101 011001 011101

B = 1010 0111 1000 0011 0010 0100 0010 1001

P(B) = $f(R_{15}, K_{16})$ = 1100 1000 1100 0000 0100 1111 1001 1000

R_{16} = 0000 1010 0100 1100 1101 1001 1001 0101.

Menurut persamaan 2.5, output putaran ke-16 dari enkripsi DES data di atas adalah,

$(L_{16}, R_{16}) = (R_{15}, L_{15} \oplus f(R_{15}, K_{16}))$.

Selanjutnya dapat diperoleh pra-*ciphertext* dari proses enkripsi algoritma DES yaitu :

(R_{16}, L_{16}) = 0000 1010 0100 1100 1101 1001 1001 0101 0100 0011 0100 0010 0011 0010 0011 0100.

Proses berikutya melakukan permutasi akhir terhadap pra-*ciphertext* tersebut.

4. Permutasi Terakhir

Setelah dilakukan 16 kali putaran terhadap blok kiri *(L)* dan blok kanan *(R)* dihasilkan pra-*ciphertext* (R_{16}, L_{16}). Untuk mendapatkan *chipertext* yang sebenarnya dilakukan permutasi akhir terhadap (R_{16}, L_{16}) menggunakan *invers* permutasi awal (IP^{-1}) seperti pada Tabel 2.7, diperoleh,

IP^{-1} = 1000 0101 1110 1000 0001 0011 0101 0100 0000 1111 0000 1010 1011 0100 0000 0101.

IP^{-1} merupakan *chipertext* dari proses enkripsi DES. Jika IP^{-1} dikonversi kedalam bilangan hexadesimal menggunakan Tabel 2.1, diperolch *chipertext* C = 85E813540F0AB405.

Jadi pesan Ahmad yang berupa *plaintext* P = 0123456789ABCDEF setelah dienkripsi dengan algoritma DES menggunakan kunci eksternal K=133457799BBCDFF1, diperoleh *ciphertext* C = 85E813540F0AB405.

5. Dekripsi DES

Proses dekripsi terhadap suatu *ciphertext* merupakan kebalikan dari proses enkripsi. Dalam hal ini DES menggunakan algoritma yang sama untuk melakukan proses enkripsi dan dekripsi. Agar Ibrahim dapat membaca pesan berupa *ciphertext* yang dikirimkan oleh Ahmad, Ibrahim harus mempunyai kunci eksternal yang sama seperti kunci yang digunakan Ahmad untuk melakukan enkripsi. Selanjutnya langkah-langkah yang harus dilakukan Ibrahim sebagai berikut.

Ciphertext yang diterima oleh Ibrahim dari Ahmad terlebih dahulu dikonversi kedalam bilangan biner menggunakan Tabel 2.1, diperoleh,

C = 85E813540F0AB405

C = 1000 0101 1110 1000 0001 0011 0101 0100 0000 1111 0000 1010 1011 0100 0000 0101.

Untuk mendapatkan $(R_{16}L_{16})$, dilakukan permutasi terhadap C menggunakan matriks *invers* permutasi awal seperti pada Tabel 2.7 sehingga diperoleh,

(R_{16},L_{16}) = 0000 1010 0100 1100 1101 1001 1001 0101 0100 0011 0100 0010 0011 0010 0011 0100.

(R_{16},L_{16}) merupakan blok input awal untuk proses dekripsi. Selanjutnya (R_{16},L_{16}) dibagi menjadi dua bagian yaitu blok kanan (R_{16}) dan blok kiri (L_{16}) yang masing-masing panjangnya 32 bit.

R_{16} = 0000 1010 0100 1100 1101 1001 1001 0101.

L_{16} = 0100 0011 0100 0010 0011 0010 0011 0100.

Selanjutnya kedua blok ini diproses kedalam 16 putaran DES sebagaimana dalam proses enkripsi. Algoritma yang digunakan dalam proses dekripsi sama seperti algoritma yang digunakan dalam proses enkripsi, hanya saja urutan kunci internal yang digunakan dibalik. Jika pada proses enkripsi urutan kunci internal yang digunakan adalah $K_1, K_2, \ldots, K_{16}$, maka pada proses dekripsi urutan kunci yang digunakan adalah $K_{16}, K_{15}, \ldots, K_1$.

Untuk setiap putaran dalam proses dekripsi, dihasilkan output yang secara matematis dapat dinyatakan dalam Persamaan 2.6. Menggunakan persamaan tersebut untuk i = 16 diperoleh,

K_{16} = 110010 110011 110110 001011 000011 100001 011111 110101 $R_{15} = L_{16}$ = 0100 0011 0100 0010 0011 0010 0011 0100

$L_{15} = R_{16} \oplus f(L_{16}, K_{16})$.

Untuk mendapatkan nilai dari L_{15}, terlebih dahulu dicari nilai $f(L_{16},K_{16})$ menggunakan diagram alir fungsi f dengan input R_{i-1} diganti L_i. Dalam hal ini L_{16} merupakan input tranformasi f. Selanjutnya menggunakan fungsi ekspansi, L_{16} yang panjangnya 32 bit akan diperluas menjadi blok yang panjangnya 48 bit dengan matriks permutasi ekspansi dalam Tabel 2.6, diperoleh $E(L_{16})$.

L_{16} = 0100 0011 0100 0010 0011 0010 0011 0100

$E(L_{16})$ = 001000 000110 101000 000100 000110 100100 000110 101000.

Selanjutnya hasil ekspansi, yaitu $E(L_{16})$, di-XOR-kan dengan kunci internal K_{16} menghasilkan vektor A yang panjangnya 48 bit. Dengan Persamaan 2.2 diperoleh vektor A sebagai berikut,

K_{16} = 110010 110011 110110 001011 000011 100001 011111 110101

E(L_{16}) = 001000 000110 101000 000100 000110 100100 000110 101000

$E(L_{16}) \oplus K_{16} = A$ = 111010 110101 011110 001111 000101 000101 011001 011101.

Vektor A yang telah dikelompokkan menjadi delapan kelompok dan masing-masing kelompok terdiri dari enam bit menjadi input untuk proses subtitusi S-Box.

A_1= 111010, A_2=110101,....,A_7= 011001, A_8=011101

Dengan Persamaan 2.3 untuk $1 \leq j \leq 8$ diperolch,

$B_1 = S_1(A_1) = S_1(r, c) = S_1(10, 1101) = S_1(2, 13) = 10 = 1010$

$B_2 = S_2(A_2) = S_2(r, c) = S_2(11, 1010) = S_2(3, 10) = 7 = 0111$

$B_3 = S_3(A_3) = S_3(r, c) = S_3(00, 1111) = S_3(0, 15) = 8 = 1000$

$B_4 = S_4(A_4) = S_4(r, c) = S_4(01, 0111) = S_4(0, 7) = 3 = 0011$

$B_5 = S_5(A_5) = S_5(r, c) = S_5(01, 0010) = S_5(0, 2) = 2 = 0010$

$B_6 = S_6(A_6) = S_6(r, c) = S_6(01, 0010) = S_6(1, 2) = 4 = 0100$

$B_7 = S_7(A_7) = S_7(r, c) = S_7(01, 1100) = S_7(1, 12) = 2 = 0010$

$B_8 = S_8(A_8) = S_8(r, c) = S_8(01, 1110) = S_8(1, 14) = 9 = 1001.$

Sehingga diperoleh output dari proses S-Box yaitu

$B = B_1B_2B_3B_4B_5B_6B_7B_8$ = 1010 0111 1000 0011 0010 0100 0010 1001.

Selanjutnya vektor B ini menjadi input untuk proses permutasi menggunakan matriks permutasi seperti Tabel 2.7 sehingga dihasilkan

$P(B)$ = 1100 1000 1100 0000 0100 1111 1001 1000.

$P(B)$ merupakan output dari fungsi f. Dalam hal ini $P(B) = f(L_i, K_i)$ sehingga untuk i=16, $P(B) = f(L_{16}, K_{16})$. Selanjutnya untuk mendapatkan L_{15} dilakukan dengan meng-XOR-kan $f(L_{16}, K_{16})$ dengan R_{16} sehingga diperoleh L_{15} sebagai berikut,

R_{16} = 0000 1010 0100 1100 1101 1001 1001 0101

$f(L_{16}, K_{16})$ = 1100 1000 1100 0000 0100 1111 1001 1000

$R_{16} \oplus f(L_{16}, K_{16})$ = 1100 0010 1000 1100 1001 0110 0000 1101.

Jadi, $L_{15} = R_{16} \oplus f(L_{16}, K_{16})$ = 1100 0010 1000 1100 1001 0110 0000 1101.

Proses di atas (untuk i=16) merupakan putaran pertama proses dekripsi menggunakan algoritma DES. Selanjutnya dengan cara yang sama dilakukan putaran ke-15 (i=15) dan seterusnya sampai dengan putaran ke-1 (i=1).

Untuk putaran ke-1 dari algoritma DES diperoleh data sebagai berikut,

K_1 – 000110 110000 001011 101111 111111 000111 000001 110010

$R_0 = L_1$ = 1111 0000 1010 1010 1111 0000 1010 1010

$E(L_1)$ = 011110 100001 010101 010101 011110 100001 010101 010101

$E(L_1) \oplus K_1 = A$ = 011000 010001 011110 111010 100001 100110 010100 100111

B = 0101 1100 1000 0010 1011 0101 1001 0111.

$P(B) = f(L_1, K_1)$ = 0010 0011 0100 1010 1010 1001 1011 1011

L_0 = 1100 1100 0000 0000 1100 1100 1111 1111.

Output putaran ke-i dari proses dekripsi DES adalah,

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

sehingga untuk putaran terakhir (i=1), output dari proses dekripsi adalah,

$(R_0, L_0) = (L_1, R_1 \oplus f(L_1, K_1))$.

Dari data di atas dapat diperoleh pra-*plaintext* dari proses dekripsi algoritma DES yaitu :

(L_0, R_0) = 1100 1100 0000 0000 1100 1100 1111 1111 1111 0000 1010 1010 1111 0000 1010 1010.

Proses selanjutnya adalah dengan melakukan permutasi akhir terhadap pra-*ciphertext* tersebut menggunakan matriks permutasi awal (IP) seperti pada Tabel 2.2 sehingga diperoleh,

IP = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111.

IP merupakan *plaintext* dari proses dekripsi DES. Jika IP dikonversi kedalam bilangan hexadesimal menggunakan Tabel 2.1, akan diperoleh *plaintext* P =0123456789ABCDEF.

Jadi, pesan yang diterima Ibrahim dari Ahmad setelah dilakukan dekripsi dengan algoritma DES menggunakan kunci eksternal K yang sama diperoleh *plaintext* P =0123456789ABCDEF.

3.1.3 Sistem Yang Akan Dibangun

Adapun sistem yang akan dibangun harus mampu melakukan fungsi sebagai berikut:

1. Sistem harus dapat menerima inputan pesan bertipe teks, dan inputan nomor telepon penerima pesan.
2. Sistem mampu membuat pesan teks yang terenkripsi dengan metode kriptografi algoritma DES dan Blowfish.
3. Sistem mampu membaca pesan terenkripsi yang tersimpan secara *default* pada aplikasi perpesanan di smartphone berbasis android.
4. Menu kirim pesan menampilkan pemberitahuan apakah pesan tersebut terkirim atau tidak terkirim.

3.1.4 Struktur Umum Sistem

Gambaran sistem secara keseluruhan dari Aplikasi SMS Kriptografi ini menguraikan proses dan aliran kerja dari aplikasi itu sendiri. Pembuatan aplikasi ini mampu menampilkan menu-menu serta melakukan proses kirim dan baca pesan SMS.

Struktur yang dimaksud adalah isi atau komponen-komponen yang terdapat pada sistem secara keseluruhan yaitu:

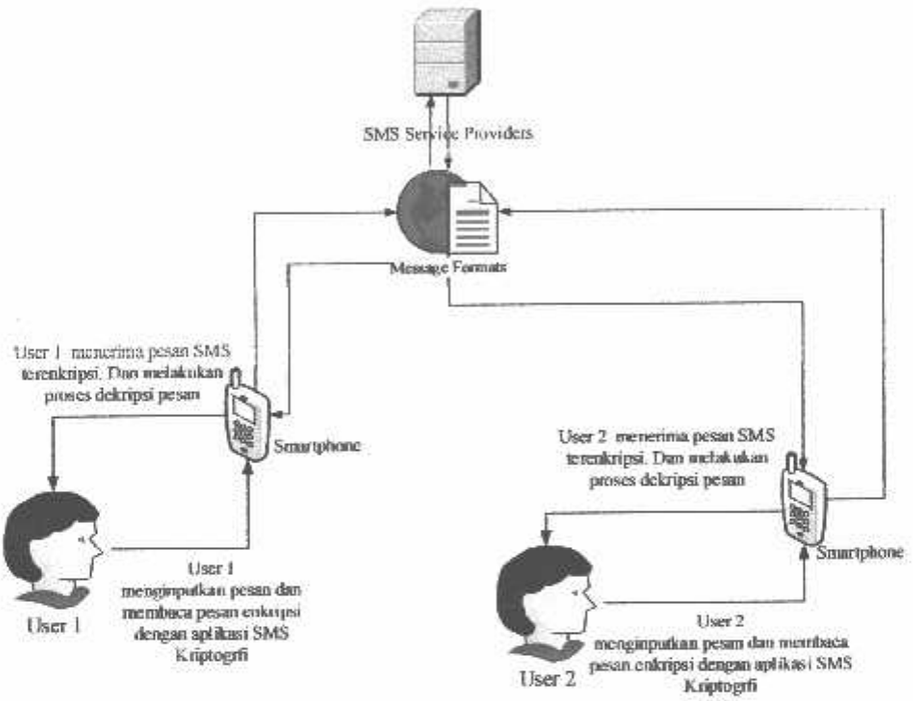
1. Tulis Pesan
   a. Input nomer telepon
   b. Input pesan
   c. Input key (kunci)
   d. Enkripsi pesan
2. Baca Pesan
   a. Input key (kunci)
   b. Dekripsi pesan
   c. Pesan masuk
3. Bantuan
4. Tentang
5. Keluar

3.2 Perancangan Sistem

Dalam perancangan aplikasi kriptografi berbasis android menggunakan algoritma DES dan Blowfish menjelaskan desain sistem, *flowchart* yang berisi rancangan *interface* aplikasi dan proses-proses interaksi *user* dengan aplikasi.

3.2.1 Desain Sistem

Sistem yang akan dibuat yaitu sebuah system kriptografi dengan SMS yang berbasis *mobile*, sehingga dapat digunakan oleh *user* untuk mengamanankan pesan dan membaca pesan yang terenkripsi seperti pada Gambar 3.1.

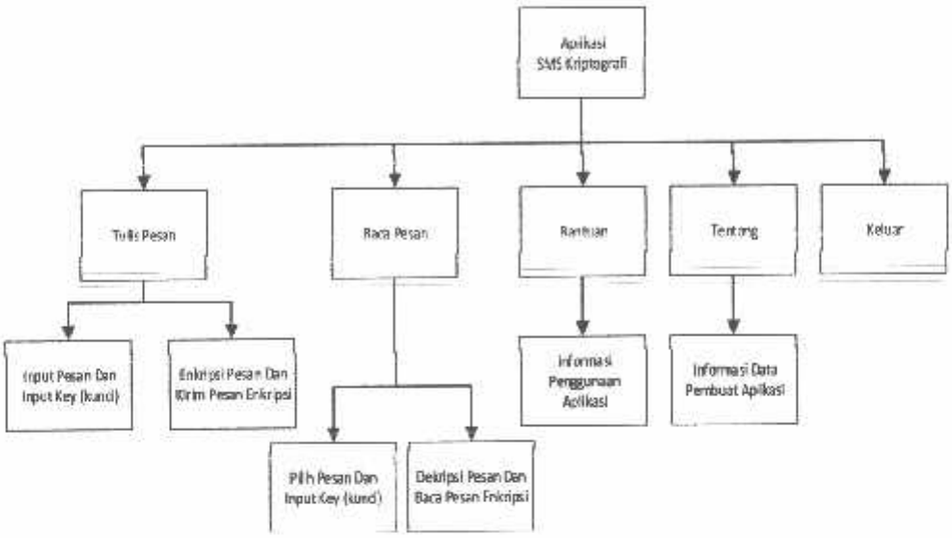


Gambar 3.1 Desain sistem aplikasi kriptografi

Pada Gambar 3.1 menjelaskan tentang desain sistem aplikasi SMS kriptografi. Dimana sistem berjalan pada perangkat *mobile* smartphone yang berbasis android. Sistem SMS kriptografi ini berjalan layaknya aplikasi SMS pada umumnya, yang membedakan adalah adanya sebuah proses enkripsi dan dekripsi untuk pengamanan pesan. Ketika *user* pertama menginputkan pesan dan mengenkripsinya maka penerima pesan akan menerima pesan yang terkodekan, untuk membaca isi pesan asli tersebut *user* sebagai penerima pesan harus

mendekripsi pesan tersebut dengan *key* atau kunci yang ditentukan pengirim pesan.

3.2.2 Sturktur Menu



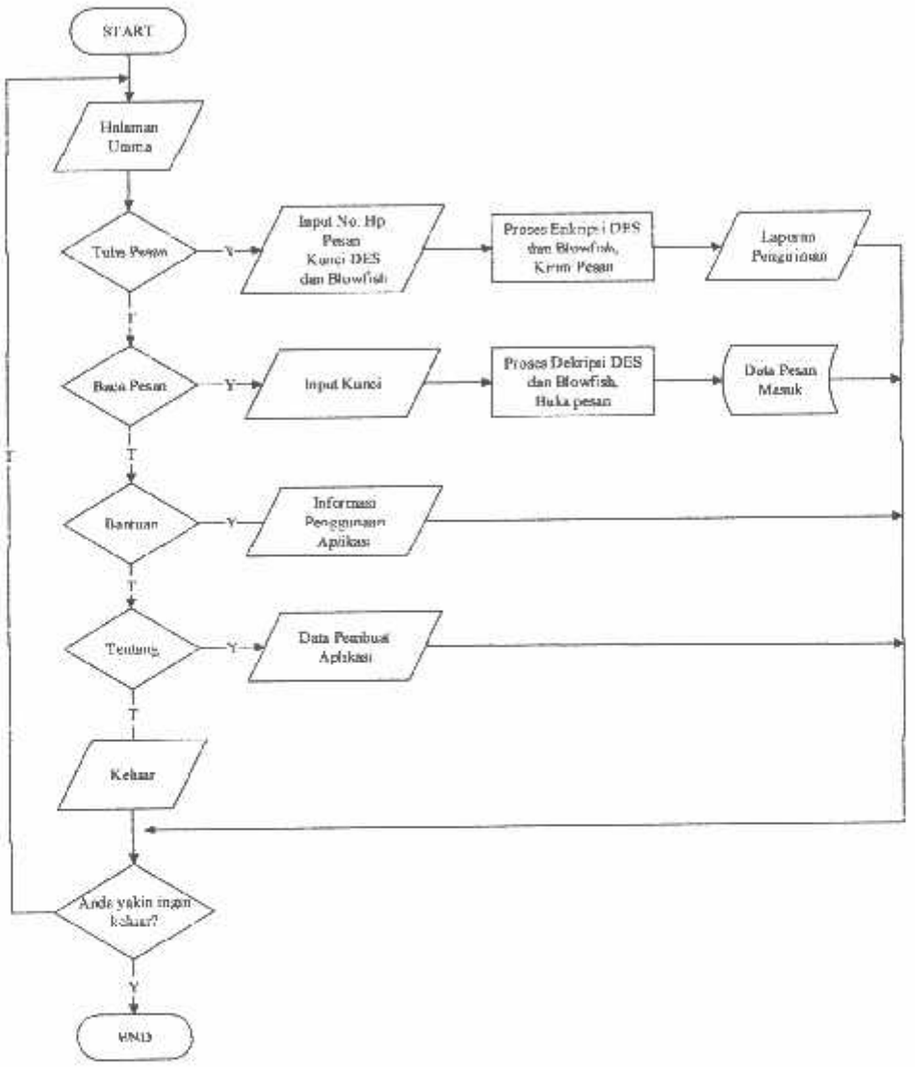
Gambar 3.2 Struktur menu aplikasi kriptografi

Pada Gambar 3.2 menjelaskan tentang struktur menu aplikasi SMS kriptografi. Menu utama terdiri dari 5 menu yaitu: Tulis pesan, Baca pesan, Bantuan, Tentang, dan menu *exit* atau keluar dari aplikasi. Pada menu tulis pesan terdapat *form* untuk menulis pesan, menginputkan nomor tujuan penerima pesan, mengenkripsi pesan dengan *key* (kunci) yang telah di inputkan dan mengirim pesan enkripsi tersebut. Pada menu baca pesan akan menampilkan *form* untuk mendekripsi atau membuka pesan asli yang terenkripsi dengan memasukkan *key* (kunci) yang dipakai saat proses enkripsi. Pada menu bantuan berisi panduan atau cara penggunaan aplikasi tersebut. Dan pada menu tentang berisi form yang menampilkan data diri pembuat aplikasi tersebut, serta menu *exit* untuk keluar dari aplikasi.

3.2.3 Flowchart

3.2.3.1 *Flowchart* Program Aplikasi Kriptografi

Berikut ini adalah flowchart program aplikasi kriptografi yang terdapat beberapa menu yang masing-masing memiliki fungsi tersendiri, terlihat seperti pada Gambar 3.3:



Gambar 3.3 Flowchart aplikasi kriptografi

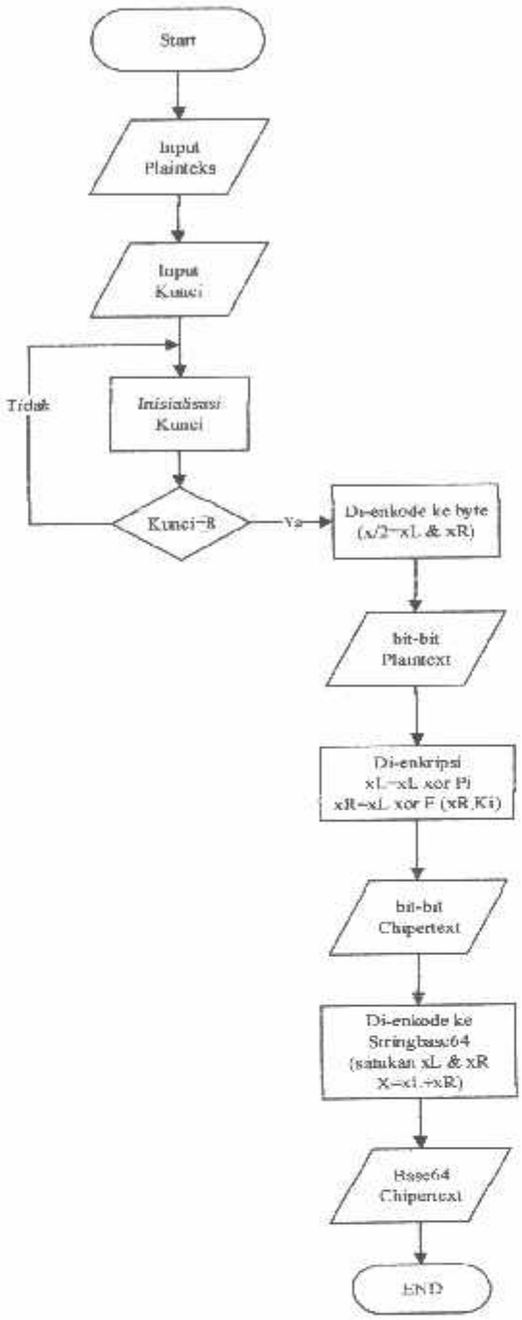
Keterangan :

1. Start: langkah pertama untuk memulai aplikasi, setelah aplikasi dijalankan akan muncul tampilan menu utama pada aplikasi seperti menu tulis pesan, baca pesan, bantuan, tentang, dan exit.

2. Menu tulis pesan: jika ya maka akan menampilkan *layout* baru yang terdiri dari beberapa inputan, seperti input nomer telepon penerima SMS, input pesan, input kunci DES dan blowfish serta ada beberapa button. Berupa button untuk enkripsi menggunakan algoritma DES, blowfish, dan gabungan kedua algoritma tersebut. Kemudian button kirim untuk mengirim pesan enkripsi gabungan kedua metode.
3. Menu baca pesan: jika ya menampilkan *layout* yang digunakan untuk dekripsi pesan yang terenkripsi dengan memasukkan kunci blowfish dan DES yang sama saat proses enkripsi yang dilakukan oleh pengirim. Pesan asli akan muncul setelah memilih salah satu chipertext DES untuk dilakukan proses dekripsi.
4. Menu bantuan: jika ya maka akan menampilkan tulisan dan gambar dari cara penggunaan menu tulis pesan dan baca pesan bagi yang belum mengerti cara untuk enkripsi dan dekripsi pesan.
5. Menu tentang: jika ya menampilkan informasi pembuat aplikasi.
6. Menu exit: keluar dari aplikasi yang dijalankan.
7. End: untuk mengakhiri aplikasi.

3.2.3.2 *Flowchart* Prosedur Enkripsi dan Dekripsi DES

Proses kerja untuk enkripsi dan dekripsi DES menggunakan kunci dan proses yang serupa. Hanya berbeda pada inputan pesannya *(plaintext-chipertext)* digunakan dengan urutan terbalik dan di inverskan. *Flowchart* program untuk proses enkripsi ditunjukan pada gambar 3.4.

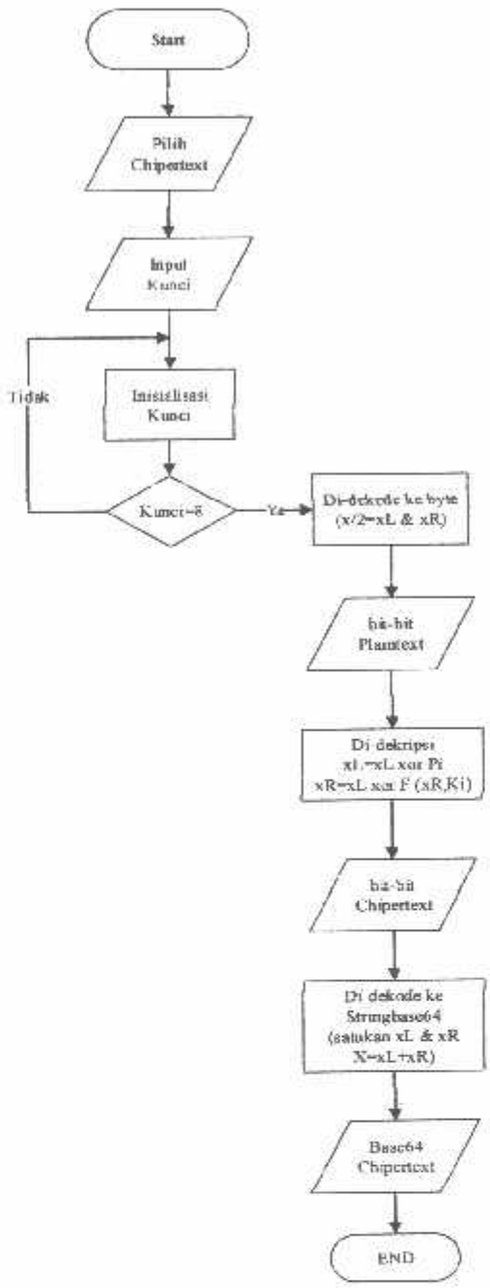


Gambar 3.4 Flowchart prosedur enkripsi DES

Keterangan:

1. Inputan pesan (x) plaintext yang akan dienkripsi diasumsikan sebagai masukan, plaintext tersebut diambil sebanyak 64-bit.

2. Inisialisasi kunci eksternal DES 8 karakter atau 64-bit, dan akan diambil 56-bit untuk proses enkripsi dan dekripsinya nanti, apabila kurang 8 karakter akan menampilkan pesan error, dan system berjalan ulang.
3. Pesan tersebut dienkode kedalam bit, x/2 =xL & xR: hasil pengambilan tadi dibagi 2, 32-bit pertama disebut xL, 32-bit kedua disebut xR.
4. Menghasilkan bit-bit *plaintext.*
5. Kemudian bit-bit plaintext dienkripsi, lakukan operasi xL= xL xor Pi dan xR = xL xor F (xR,Ki)
6. Menghasilkan bit-bit chipertext, hasil dari operasi diatas ditukar xL menjadi xR dan xR menjadi xL, lakukan sebanyak 16 kali, perulangan yang ke-16 lakukan lagi penukaran xL dan xR.
7. Selanjutnya di-enkode ke stringbase64, x = xL+xR agar bit-bit tersebut tetap dapat dipetakan dalam karakter-karakter (.txt).
8. Dan menghasilkan chipertext base64.

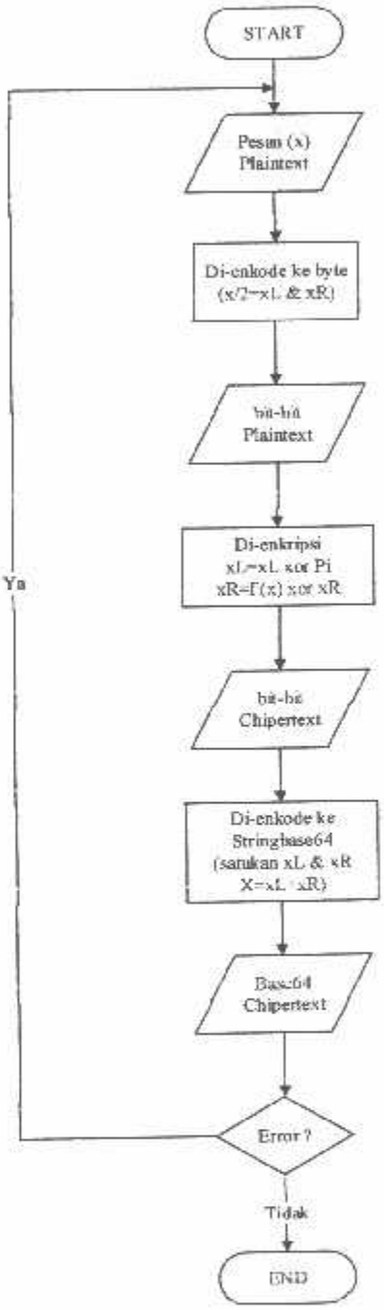


Gambar 3.5 Flowchart prosedur dekripsi DES

Flowchart program untuk proses dekripsi DES hampir sama dengan proses enkripsi bedanya inputan pesannya *(plaintext-chipertext)* digunakan dengan urutan terbalik, dipermutasi dengan matriks permutasi balik (invers initial permutation atau IP- 1) menjadi blok teks-kode. IP^{-1} ke bitstring $R_{16}L_{16}$, memperoleh teks-kode y, kemudian $y = IP^{-1}$ $(R_{16}L_{16})$ seperti ditunjukan pada gambar 3.5.

3.2.3.3 *Flowchart* Prosedur Enkripsi dan Dekripsi Blowfish

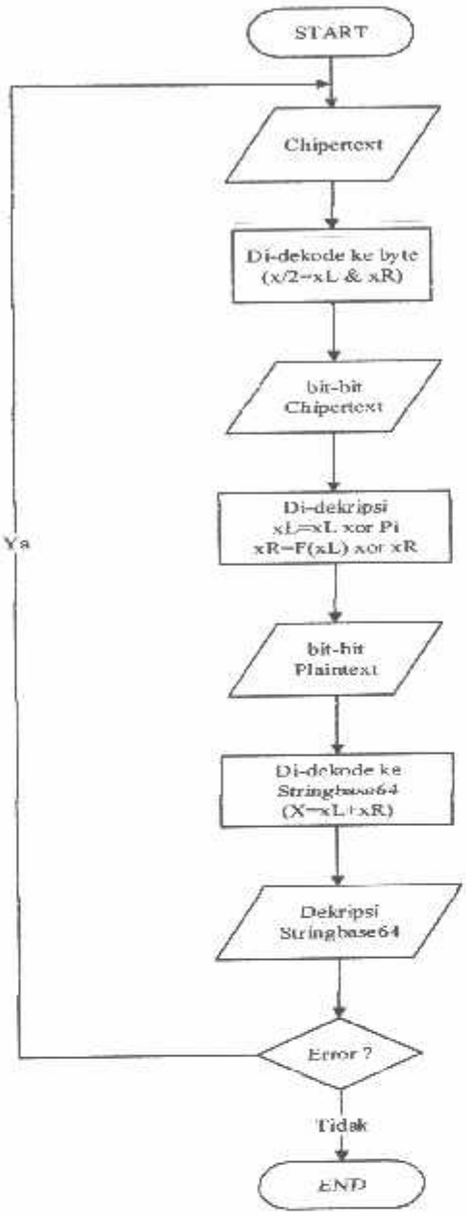
Proses kerja untuk enkripsi dan dekripsi blowfish menggunakan kunci dan proses yang serupa. Hanya berbeda pada inputan pesannya *(plaintext-chipertext)* digunakan dengan urutan terbalik dan di inverskan. *Flowchart* program untuk proses enkripsi ditunjukan pada gambar 3.6.



Gambar 3.6 Flowchart prosedur enkripsi blowfish

Keterangan:

1. Inputan pesan (x) plaintext yang akan dienkripsi diasumsikan sebagai masukan, plaintext tersebut diambil sebanyak 64-bit dan apabila kurang dari 64-bit maka kita tambahkan bit-nya, supaya dalam operasi nanti sesuai datanya.
2. Pesan tersebut dienkode kedalam bit, x/2 =xL & xR: hasil pengambilan tadi dibagi 2, 32-bit pertama disebut xL, 32-bit kedua disebut xR.
3. Menghasilkan bit-bit *plaintext*.
4. Kemudian bit-bit plaintext dienkripsi, lakukan operasi xL= xL xor Pi dan xR = F(xL) xor xR.
5. Menghasilkan bit-bit chipertext, hasil dari operasi diatas ditukar xL menjadi xR dan xR menjadi xL, lakukan sebanyak 16 kali, perulangan yang ke-16 lakukan lagi penukaran xL dan xR.
6. Pada proses ke-17 lakukan operasi untuk xR = xR xor P17 dan xL = xL xor P18.
7. Selanjutnya di-enkode kc stringbase64, x = xL+xR agar bit-bit tersebut tetap dapat dipetakan dalam karakter-karakter (.txt).
8. Dan menghasilkan chipertext base64.

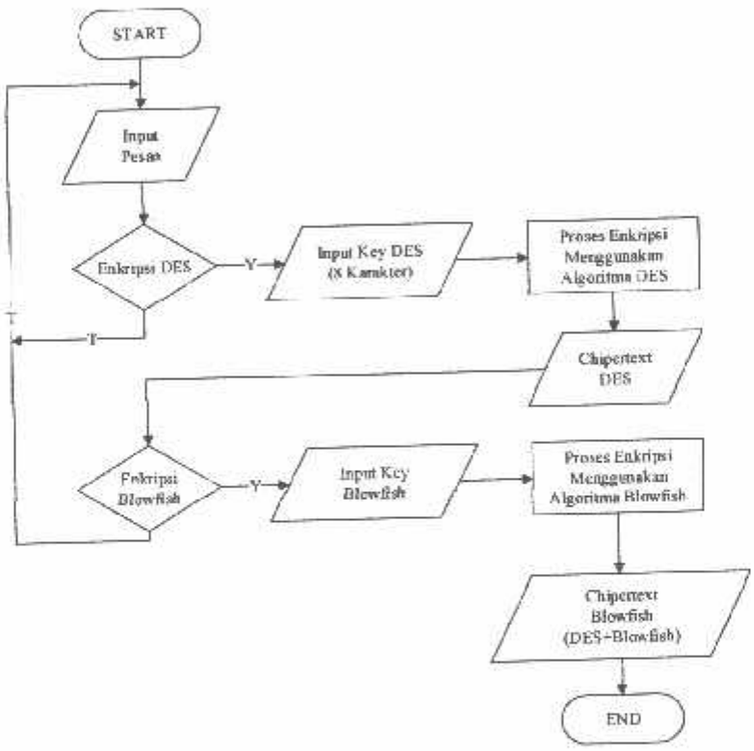


Gambar 3.7 Flowchart prosedur dekripsi blowfish

Flowchart program untuk proses dekripsi blowfish hampir sama dengan proses enkripsi bedanya inputan pesannya *(plaintext-chipertext)* digunakan dengan urutan terbalik, seperti ditunjukan pada gambar 3.7.

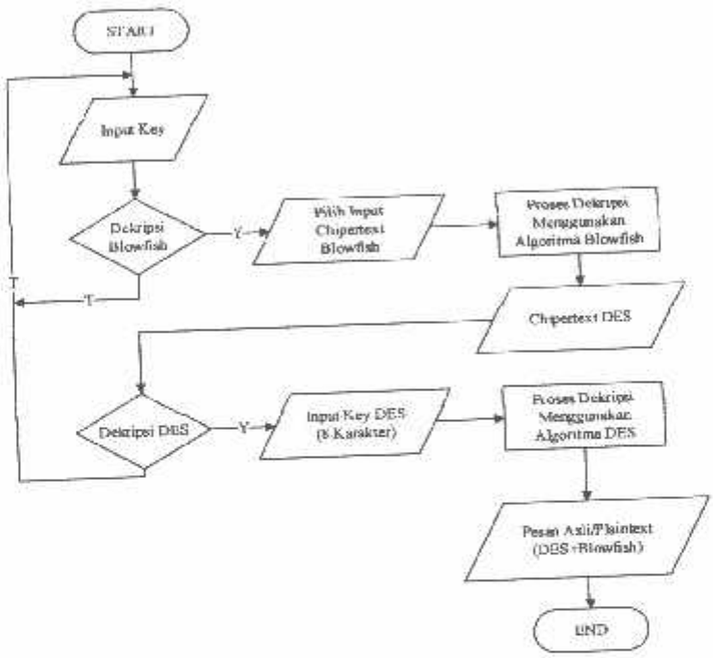
3.2.3.4 *Flowchart* Enkripsi dan Dekripsi Gabungan DES Dan Blowfish

Adapun untuk *flowchart* enkripsi gabungan DES dan blowfish pertama adalah menginputkan pesan yang dijadikan sebagai inputan plainteks, kemudian masukkan kunci DES 8 karakter untuk selanjutnya dilakukan proses enkripsi dengan menggunakan algoritma DES dan akan menghasilkan chiperteks DES. Chiperteks DES digunakan untuk enkripsi selanjutnya yaitu dengan menggunakan algoritma blowfish, chiperteks DES dijadikan plainteks untuk blowfish dan langsung dienkripsi menggunakan kunci blowfish. Maka akan menghasilkan chiperteks gabungan dari algoritma DES dan blowfish. *Flowchart* untuk enkripsi gabungan seperti ditunjukan pada Gambar 3.8.



Gambar 3.8 Flowchart enkripsi gabungan DES dan blowfish

Untuk *flowchart* dekripsi gabungan DES dan blowfish pertama adalah menginputkan key pertama yaitu key blowfish yang digunakan enkripsi plainteks, kemudian dilakukan proses dekripsi menggunakan algoritma blowfish selanjutnya hasil dekripsi menghasilkan chiperteks DES. Pilih ciperteks DES untuk dekripsi selanjutnya, yaitu dengan menggunakan algoritma DES. Chiperteks DES terpilih akan didekripsi lagi untuk mengetahui pesan asli atau plainteks dengan menggunakan kunci 8 karakter saat proses enkripsi. Maka pesan asli dari enkripsi gabungan dapat dipecahkan dan diketahui isinya. *Flowchart* untuk dekripsi gabungan terlihat seperti pada Gambar 3.9.



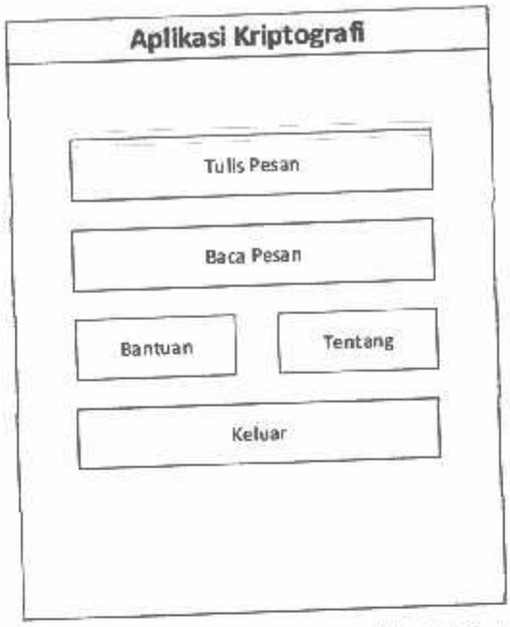
Gambar 3.9 Flowchart dekripsi gabungan DES dan blowfish

3.2.4 Desain Layout

Berikut ini adalah desain layout dari program yang akan dibuat sebagai berikut:

1. Halaman Utama

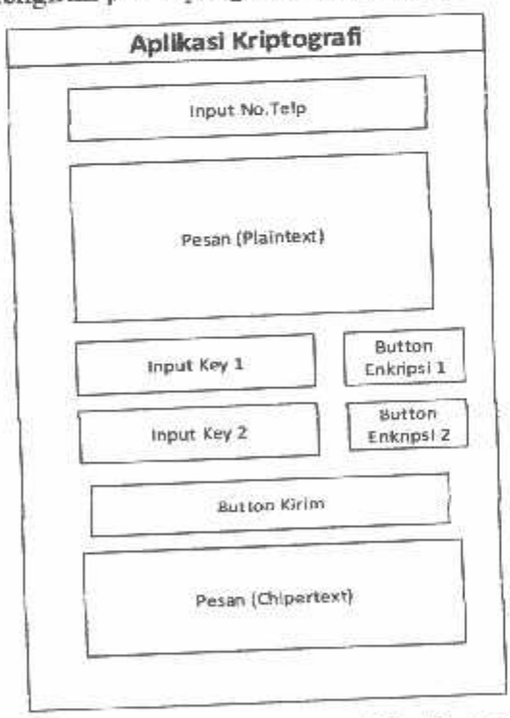
Pada halaman ini menampilkan menu pertama yang muncul setelah aplikasi dibuka yang terdapat menu tulis pesan, baca pesan, bantuan, tentang, dan *exit* atau keluar.



Gambar 3.10 Halaman Utama aplikasi kriptografi

2. Menu Tulis Pesan

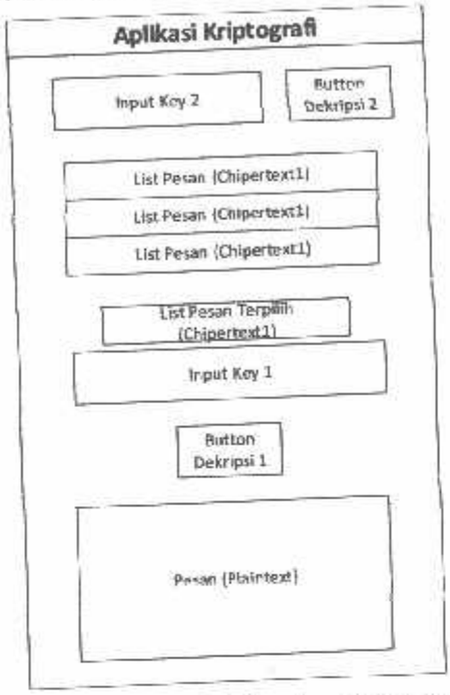
Pada halaman tulis pesan menampilkan *form* untuk menginputkan nomor telepon penerima pesan, menulis pesan, menginputkan *key* (kunci) untuk mengenkripsi pesan dan mengirim pesan yang telah terenkripsi.



Gambar 3.11 Menu Tulis Pesan aplikasi kriptografi

3. Menu Kotak Masuk

Pada halaman menu ini menampilkan *layout* untuk membaca pesan yang terenkripsi dengan menginputkan *key* (kunci) yang dipakai saat proses enkripsi pesan, sehingga pesan asli dapat terbaca oleh penerima pesan.



Gambar 3.12 Menu Kotak Masuk aplikasi kriptografi

4. Menu Bantuan

Pada menu bantuan menampilkan informasi bantuan penggunaan aplikasi berupa gambar dari *screenshot* aplikasi yang diberi penjelasan pada masing-masing komponen.

BAB IV

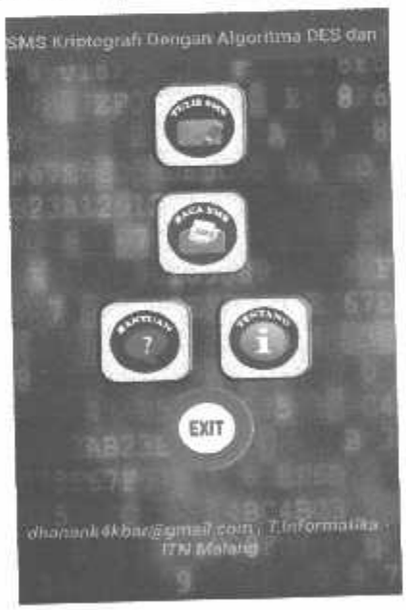
IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi Hasil

Tahapan hasil akhir dari aplikasi penerapan kriptografi dengan algoritma DES *(Data Encryption Standard)* dan blowfish pada SMS berbasis android ini dalam bentuk implementasi. Tampilan yang terdapat pada aplikasi penerapan kriptografi dengan algoritma DES *(Data Encryption Standard)* dan blowfish pada SMS berbasis android yaitu tampilan menu utama, tampilan menu tulis pesan, tampilan menu baca pesan, tampilan menu tentang, dan tampilan menu bantuan. Implenetasi hasil ini bertujuan untuk menerapkan perancangan yang akan berisi masukan-masukan demi dikembangkannya aplikasi penerapan kriptografi dengan algoritma DES *(Data Encryption Standard)* dan blowfish pada SMS berbasis android.

4.1.1 Tampilan Menu Utama

Tampilan utama adalah tampilan awal yang yang muncul setelah aplikasi dibuka yang terdapat menu tulis pesan, baca pesan, bantuan, tentang, dan *exit* atau keluar. Dapat dilihat pada Gambar 4.1.



Gambar 4.1 Tampilan menu utama

4.1.2 Tampilan Menu Tulis Pesan

Tampilan tulis pesan menampilkan *form* untuk menginputkan nomor telepon penerima pesan, menulis pesan, menginputkan *key* (kunci) untuk mengenkripsi pesan dan mengirim pesan yang telah terenkripsi. Dapat dilihat pada Gambar 4.2.

Gambar 4.2 Tampilan menu tulis pesan

4.1.3 Tampilan Menu Baca Pesan

Tampilan menu baca pesan menampilkan *form* untuk membaca pesan yang terenkripsi dengan menginputkan *key* (kunci) yang dipakai saat proses enkripsi pesan, sehingga pesan asli dapat terbaca oleh penerima pesan. Dapat dilihat pada Gambar 4.3.

Gambar 4.3 Tampilan menu baca pesan

4.1.4 Tampilan Menu Bantuan

Tampilan menu bantuan menampilkan informasi bantuan penggunaan aplikasi berupa gambar dari *screenshot* aplikasi yang diberi penjelasan pada masing-masing komponen. Dapat dilihat pada Gambar 4.4.

Gambar 4.4 Tampilan menu bantuan

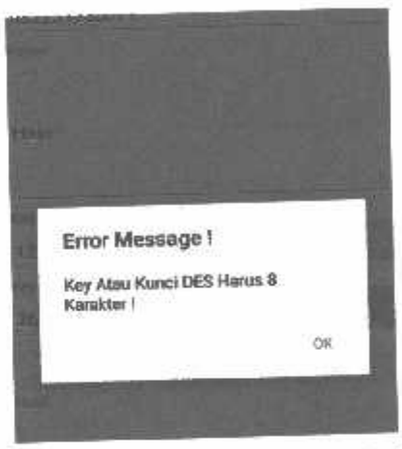
4.1.5 Tampilan Menu Tentang

Tampilan menu tentang berisi informasi dari pembuat aplikasi yang terdiri dari gambar profil pembuat, nama institut, dan data diri lainnya. Dapat dilihat pada Gambar 4.5.

Gambar 4.5 Tampilan menu tentang

4.1.6 Tampilan *Alert Dialog*

Tampilan ini merupakan tampilan pesan peringatan apabila terjadi kesalahan input yang dilakukan oleh pengguna aplikasi. Seperti kesalahan saat memasukan kunci untuk enkripsi dengan algoritma DES. Dapat dilihat pada Gambar 4.6.



Gambar 4.6 Tampilan alert dialog

4.1.7 Tampilan Hasil Enkripsi

Tampilan hasil enkripsi pesan menggunakan algoritma DES dan blowfish dengan menggunakan kunci DES 8 karakter dan menggunakan kunci blowfish dengan panjang dari 32 bit hingga 48 bit. Terlihat seperti Gambar 4.7.

Gambar 4.7 Tampilan hasil enkripsi

4.1.8 Tampilan Hasil Dekripsi

Tampilan untuk dekripsi pesan menggunakan kunci algoritma DES dengan panjang 8 karakter yang dipakai saat enkripsi pesan, begitu pula dengan kunci algoritma blowfish yang dipakai saat enkripsi pesan. Dapat dilihat pada Gambar 4.8.

Gambar 4.8 Tampilan hasil dekripsi

4.2 Pengujian Fungsional

Dalam pengujian fungsional dalam aplikasi ini mengenai proses fungsional yang terjadi dalam aplikasi. Hasil pengujian dapat dilihat pada tabel 1.

Tabel 4.1.Pengujian Fungsional

No.	Uraian	Perangkat dan Jenis Layar					
		1		2		3	
		P	L	P	L	P	L
1	Judul aplikasi (marquee text)	✓	x	✓	x	✓	x
2	Button Tulis Pesan	✓	✓	✓	✓	✓	✓
3	Button Baca Pesan	✓	✓	✓	x	✓	✓
4	Button Bantuan	✓	✓	✓	✓	✓	✓
5	Button Tentang	✓	✓	✓	✓	✓	✓
6	Button Exit	✓	✓	✓	x	✓	✓
7	Halaman Tulis Pesan	✓	✓	✓	✓	✓	✓
8	Halaman Baca Pesan	✓	✓	✓	✓	✓	✓
9	Halaman Bantuan	✓	✓	✓	✓	✓	✓
10	Halaman Tentang	✓	x	✓	x	✓	x

Keterangan:

✓ : Berhasil

x : Gagal

P : Layar Potrait

L : Layar Landscape

Perangkat 1: Axioo PICOphone M4P M3, Quad Core 1.3 GHz Cortex-A7, RAM 2 GB, Versi Android 5.1 Lollipop, Ukuran Layar 720x1280.

Perangkat 2: OPPO Neo 3 R831K, Dual Core 1.3 GHz, RM 1 GB, Versi Android 4.2.2 Jelly Bean, Ukuran Layar 854x480.

Perangkat 3: Sony Xperia Z 2, Qualcomm Quad Core 2.3 GHz, RAM 2 GB, Versi Android 6.0 Marshmallow, Ukuran Layar 1920x1080.

Dari tabel pengujian fungsional dapat disimpulkan bahwa semua button dan tampilan *layout* pada aplikasi, fungsinya dapat berjalan sesuai dengan tujuan yang diharapkan. Namun ada beberapa button dan *layout* yang tidak sesuai fungsinya pada mode layar landscape sebagian menu tidak terdeteksi, sedangkan pada mode layar potrait semua menu pada aplikasi sesuai dengan fungsinya.

4.3 Pengujian *User*

Setelah dilakukan pengujian aplikasi proses selanjutnya dilakukan pengujiann kepada pengguna untuk mengetahui pendapat atau penilaian pengguna aplikasi yang di bagi menjasi 6 kategori. Adapun pengujian *user* untuk aplikasi ini didasarkan pada beberapa pertanyaan yang berhubungan dengan permasalahan pada aplikasi.

Tabel 4.2.Pengujian *User*

No.	Pertanyaan	Jumlah yang menjawab	
		Ya	Tidak
1	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish dapat berjalan pada versi android Anda?	6	0
2	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish tampilannya menarik?	5	1

3	Apakah aplikasi SMS kriptografi ini mudah dioperasikan?	4	2
4	Apakah aplikasi SMS kriptografi ini bermanfaat untuk mengamankan pesan?	6	0
5	Apakah proses enkripsi pesan berhasil pada aplikasi SMS kriptografi ini?	6	0
6	Apakah proses dekripsi pesan berhasil pada aplikasi SMS kriptografi ini?	6	0

Dari data kuisioner diatas terhadap 6 responden dapat disimpulkan bahwa:

1. 100% responden menjawab **Ya** bahwa aplikasi dapat berjalan pada versi android pengguna, 0% menjawab **Tidak** berjalan pada versi android pengguna.
2. 90% responden menjawab **Ya** bahwa aplikasi memiliki tampilan menarik, 10% menjawab **Tidak** menarik pada tampilan aplikasi.
3. 80% responden menjawab **Ya** bahwa aplikasi mudah dioperasikan, 20% menjawab **Tidak** mudah dioperasikan.
4. 100% responden menjawab **Ya** bahwa aplikasi bermanfaat mengamankan pesan, 0% menjawab **Tidak** bermanfaat.
5. 100% responden menjawab **Ya** bahwa enkripsi pesan berhasil, 0% menjawab **Tidak** behasil mengenkripsi pesan.
6. 100% responden menjawab **Ya** bahwa dekripsi pesan berhasil, 0% menjawab **Tidak** behasil mendekripsi pesan.

BAB V
PENUTUP

5.1 Kesimpulan

Akhir dari penyelesaian aplikasi penerapan kriptografi dengan algoritma DES *(data encryption standard)* dan blowfish pada SMS dapat diambil kesimpulan:

1. Aplikasi dapat berjalan dengan baik pada perangkat android dengan OS jellybean, lollipop, dan marshmallow.
2. Berdasarkan pengujian user yang dilakukan bahwa aplikasi dapat melakukan proses kirim pesan enkripsi dan dekripsi pesan.
3. Berdasarkan pengujian fungsional yang dilakukan bahwa semua fungsi aplikasi 100% dapat berjalan dengan baik pada perangkat dengan OS android versi 4.2.2 (jelly bean), 5.1 (lollipop), 6.0 (marshmallow).

5.2 Saran

Dari pembuatan aplikasi ini, penulis memberikan saran yaitu:

1. Untuk pengembangan aplikasi ini bisa dibuat OS lain seperti untuk IOS, dirasa perlu untuk dicoba mengingat masing-masing OS memiliki karakteristik serta kelebihannya masing-masing.
2. Fitur aplikasi ini masih sangat sederhana, diharapkan pada penelitian selanjutnya dapat dibuat fitur aplikasi yang lebih banyak, seperti menambahkan fitur broadcast SMS.


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

DAFTAR PUSTAKA

Agustinus, Aditya Eka.2009. *Teknik Pengamanan Data pada Sistem Proses Surat dengan Metode Blowfish di Badan Kepegawaian Daerah Provinsi Jawa Tengah*, skripsi tidak dipublikasikan, Universitas Dian Nuswantoro Semarang.

Ariyus, Dony.2008 *Pengantar Ilmu Kriptografi*, Yogyakarta:Andy Offset.

Hillebrand, Friedhelm. 2010. *Short Message Service* (SMS). Canada: Wiley

Kadir, Abdul.2004 *Dasar Pemrograman Java 2*, Yogyakarta:Andy Offset.

Kahate, Atul.2013. *Cryptography and Network Security Third Edition*, New Delhi, McGraw Hill Education (India).

Nicolas Gramlich: http://andbook.anddev.org diakses pada tanggal 26 april 2016.

Safaat, N. 2012. *Pemrograman Aplikasi Mobile Smartphone dan Tablet PC Berbasis Android*. Bandung : Informatika Bandung.

Salbino, Sherief. 2014. Buku Pintar Gadget Android untuk Pemula. Jakarta:Kunci Komunikasi

Sofwan Aghus 2006 APLIKASI KRIPTOGRAFI DENGAN ALGARITMA MESSAGE DIGEST 5 (MD5).Transmini Vol 11 No 1 Juni 2006:22-27.

Sulidar Fitri. 2010. *Implementasi Algoritma Kriptografi DES dan Watermark dengan LSB*. https://www.academia.edu/7653558/IMPLEMENTASI_ALGORITMA_KRIPTOGRAFI_DES_DAN_WATERMARK_DENGAN_METODE_LSB Diakses pada tanggal 23 April 2016

Suristi Sitijak, Yuli Fauziah dan Juwairiah. 2010. *Aplikasi Kriptografi File Menggunakan Algoritma Blowfish*. https://www.academia.edu/5849075/APLIKASI_KRIPTOGRAFI_FILE_MENGGUNAKAN_ALGORITMA_BLOWFISH. Diakses pada tanggal 28 April 2016

Stiawan, Deris.2005. *Sistem Keamanan Komputer*. Jakarta:PT Elex Media Komputindo.

Thorsteinson, Petter dan Arun G. Gnana. 2004. Net Security and Cryptography. United States of America:Prentice Hall PTR

Wahana Komputer. 2005. *Pengembangan Aplikasi Sistem Informasi Akademik Berbasis SMS dengan Java*. Jakarta: Salemba Infotek.

Wikipedia. 2010. *Definisi Kriptografi*. https://id.wikipedia.org/wiki/Kriptografi .Diakses pada tanggal 12 Mei 2016

Zahra Khadijah. 2013. *Aplikasi pada Ponsel Android untuk Administrasi User Jaringan Melalui Short Message Service*. skripsi tidak dipublikasikan, Universitas Muhammadiyah Surakarta.

LAMPIRAN

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
PROGRAM STUDI TEKNIK INFORMATIKA S-1
Jl. Karanglo, Km.2 Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

NAMA : Danang Nur Fadlilah Akbar
NIM : 12.18.178
JURUSAN : Teknik Informatika S-1
JUDUL : Penerapan Kriptografi Dengan Algoritma DES *(Data Ecryption Standard)* Dan Blowfish Pada SMS Berbasis Android

Dipertahankan dihadapan Majelis Penguji Skripsi Jenjang Strata Satu (S-1) pada :
Hari : Senin
Tanggal : 25 Juli 2016
Nilai : B+

Panitia Ujian Skripsi :

Ketua Majelis Penguji

Joseph Dedy Irawan, ST, MT
NIP. 197404162005011002

Anggota Penguji :

Dosen Penguji I

Ahmad Fahrudi Setiawan, S.Kom, MT
NIP.P 1031500497

Dosen Penguji II

Karina Auliasari, ST.M.Eng
NIP.P. 1031000426

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
PROGRAM STUDI TEKNIK INFORMATIKA S-1
Jl. Karanglo, Km.2 Malang

FORMULIR PERBAIKAN SKRIPSI

Dalam pelaksanaan ujian skripsi jenjang Strata 1 Program Studi Teknik Informatika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : Danang Nur Fadlilah Akbar
NIM : 12.18.178
JURUSAN : Teknik Informatika S-1
JUDUL : Penerapan Kriptografi Dengan Algoritma DES *(Data Ecryption Standard)* Dan Blowfish Pada SMS Berbasis Android

No	Penguji	Tanggal	Uraian	Paraf
1.	Penguji I	25 Juli 2016	1. Latar Belakang dijelaskan pengguna aplikasi 2. Perbedaan SMS biasa dengan aplikasi ini 3. Alasan pemilihan algoritma DES	
2.	Penguji II	25 Juli 2016	1. Kesimpulan harus sesuai pengujian	

Dosen Penguji I

Ahmad Fahrudi Setiawan, S.Kom, MT
NIP.P 1031500497

Dosen Penguji II

Karina Auliasari, ST.M.Eng
NIP.P. 1031000426

Dosen Pembimbing I

Joseph Dedy Irawan, ST.MT
NIP. 197404162005011002

Dosen Pembimbing II

Nurlaily Vendyansyah, ST
NIP.

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

Malang, 11April 2015

Nomor : ITN-705/IV.INF/TA/2016
Lampiran : —
Perihal : *Bimbingan Skripsi*

Kepada : Yth. Bpk/IbuJoseph Dedy Irawan. ST.MT
Dosen Pembina Program Studi Teknik Informatika S-1
Institut Teknologi Nasional
Malang

Dengan Hormat,
Sesuai dengan permohonan dan persetujuan dalam proposal skripsi untuk mahasiswa :

Nama : DANANG NUR FADLILAH AKBAR
Nim : 1218178
Prodi : Teknik Informatika S-1
Fakultas : Teknologi Industri

Maka dengan ini pembimbingan kami serahkan sepenuhnya kepada Saudara/i selama waktu 6 (enam) bulan, terhitung mulai tanggal :

11April2016 S/D 11Oktober 2016

Sebagai satu syarat untuk menempuh Ujian Akhir Sarjana Teknik, Program Studi Teknik Informatika S-1.
Demikian agar maklum dan atas perhatian serta bantuannya kami sampaikan terima kasih.

Mengetahui
Program Studi Teknik InformatikaS-1
K e t u a,

Joseph Dedy Irawan,ST., MT.
NIP : 197404162005021002

Form S-4a

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
PROGRAM STUDI TEKNIK INFORMATIKA S-1
Jl. Karanglo, Km.2 Malang

BUKTI BIMBINGAN SKRIPSI

Nama : Danang Nur Fadlilah Akbar
NIM : 1218178
Jurusan : Teknik Informatika S-1
Judul : Penerapan Kriptografi Dengan Algoritma DES *(Data Ecryption Standard)* Dan Blowfish Pada SMS Berbasis Android

No.	Tanggal	Uraian	Paraf Pembimbing
1.	20/05/2016	Penulisan format proposal progres	
2.	24/05/2016	Acc seminar progres	
3.	30/05/2016	Revisi program, button enkripsi dua metode	
4.	31/05/2016	Revisi laporan BAB III	
5.	06/06/2016	Revisi program, button dekripsi metode dibuat lebih mudah dimengerti	
6.	27/06/2016	Acc seminar hasil	
7.	12/07/2016	Pengujian fungsional dan pengujian *user*	
8.	15/07/2016	Revisi BAB IV dan BAB V	
9.	22/07/2016	Acc seminar kompre	

Dosen Pembimbing I

Joseph Dedy Irawan, ST.MT
NIP. 197404162005011002

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG

INSTITUT TEKNOLOGI NASIONAL MALANG

FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK

PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG

Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

Malang, 11April 2016

Nomor : ITN-705/IV.INF/TA/2016
Lampiran : —
Perihal : *Bimbingan Skripsi*

Kepada : Yth. Bpk/IbuNurlaily Vendyansyah. ST
Dosen Pembina Program Studi Teknik Informatika S-1
Institut Teknologi Nasional
Malang

Dengan Hormat,
Sesuai dengan permohonan dan persetujuan dalam proposal skripsi untuk mahasiswa :

Nama : DANANG NUR FADLILAH AKBAR
Nim : 1218178
Prodi : Teknik Informatika S-1
Fakultas : Teknologi Industri

Maka dengan ini pembimbingan kami serahkan sepenuhnya kepada Saudara/i selama waktu 6 (enam) bulan, terhitung mulai tanggal :

11April2016 S/D 11Oktober 2016

Sebagai satu syarat untuk menempuh Ujian Akhir Sarjana Teknik, Program Studi Teknik Informatika S-1.
Demikian agar maklum dan atas perhatian serta bantuannya kami sampaikan terima kasih.

Mengetahui
Program Studi Teknik InformatikaS-1
K e t u a,

Joseph Dedy Irawan,ST., MT.
NIP : 197404162005021002

Form S-4a

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
PROGRAM STUDI TEKNIK INFORMATIKA S-1
Jl. Karanglo, Km.2 Malang

BUKTI BIMBINGAN SKRIPSI

Nama : Danang Nur Fadlilah Akbar
NIM : 1218178
Jurusan : Teknik Informatika S-1
Judul : Penerapan Kriptografi Dengan Algoritma DES *(Data Ecryption Standard)* Dan Blowfish Pada SMS Berbasis Android

No.	Tanggal	Uraian	Paraf Pembimbing
1.	18/05/2016	Penulisan sesuai format dari ITN Malang	
2.	23/05/2016	Tambah perhitungan matematis metode	
3.	24/05/2016	Acc seminar progres	
4.	31/05/2016	Revisi Program terapkan metode enkripsi dan dekripsi	
5.	06/06/2016	Revisi laporan BAB II dan BAB III	
6.	25/06/2016	Abstrak, makalah seminar hasil	
7.	12/07/2016	Revisi laporan BAB IV	
8.	15/07/2016	Kesimpulan sesuai hasil pengujian	
9.	22/07/2016	Penulisan daftar pustaka dari jurnal	
10.	23/07/2016	Acc seminar kompre	

Dosen Pembimbing II

23/7/16

Nurlaily Vendyansyah, ST
NIP.

KUISIONER RANCANG BANGUN PENERAPAN KRIPTOGRAFI DENGAN ALGORITMA DES *(DATA ENCRYPTION STANDARD)* DAN BLOWFISH PADA SMS BERBASIS ANDROID

Nama : Angga Samuel Arromo

Pekerjaan : Mahasiswa

OS Android : Samsung GALAXY ACE (Lollipop)

No.	Pertanyaan	Jawaban	
		Ya	Tidak
1	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish dapat berjalan pada versi android Anda?	✓	
2	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish tampilannya menarik?	✓	
3	Apakah aplikasi SMS kriptografi ini mudah dioperasikan?	✓	
4	Apakah aplikasi SMS kriptografi ini bermanfaat untuk mengamankan pesan?	✓	
5	Apakah proses enkripsi pesan berhasil pada aplikasi SMS kriptografi ini?	✓	
6	Apakah proses dekripsi pesan berhasil pada aplikasi SMS kriptografi ini?	✓	

Catatan: jawablah masing-masing pertanyaan dengan memberikan tanda centang (√) pada salah satu jawaban Ya atau Tidak.

Pengguna

(.....)

KUISIONER RANCANG BANGUN PENERAPAN KRIPTOGRAFI DENGAN ALGORITMA DES *(DATA ENCRYPTION STANDARD)* DAN BLOWFISH PADA SMS BERBASIS ANDROID

Nama : ANDRI ARFINANDA

Pekerjaan : MAHASISWA

OS Android : JELLY BEAN

No.	Pertanyaan	Jawaban	
		Ya	Tidak
1	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish dapat berjalan pada versi android Anda?	√	
2	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish tampilannya menarik?	√	
3	Apakah aplikasi SMS kriptografi ini mudah dioperasikan?		√
4	Apakah aplikasi SMS kriptografi ini bermanfaat untuk mengamankan pesan?	√	
5	Apakah proses enkripsi pesan berhasil pada aplikasi SMS kriptografi ini?	√	
6	Apakah proses dekripsi pesan berhasil pada aplikasi SMS kriptografi ini?	√	

Catatan: jawablah masing-masing pertanyaan dengan memberikan tanda centang (√) pada salah satu jawaban Ya atau Tidak.

Pengguna

(ANDRI ARFINANDA)

KUISIONER RANCANG BANGUN PENERAPAN KRIPTOGRAFI DENGAN ALGORITMA DES *(DATA ENCRYPTION STANDARD)* DAN BLOWFISH PADA SMS BERBASIS ANDROID

Nama : Hari Gagas Kusuma

Pekerjaan : Mahasiswa

OS Android : Jelly bean

No.	Pertanyaan	Jawaban	
		Ya	Tidak
1	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish dapat berjalan pada versi android Anda?	✓	
2	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish tampilannya menarik?	✓	
3	Apakah aplikasi SMS kriptografi ini mudah dioperasikan?	✓	
4	Apakah aplikasi SMS kriptografi ini bermanfaat untuk mengamankan pesan?	✓	
5	Apakah proses enkripsi pesan berhasil pada aplikasi SMS kriptografi ini?	✓	
6	Apakah proses dekripsi pesan berhasil pada aplikasi SMS kriptografi ini?	✓	

Catatan: jawablah masing-masing pertanyaan dengan memberikan tanda centang (√) pada salah satu jawaban Ya atau Tidak.

Pengguna

(.....)

KUISIONER RANCANG BANGUN PENERAPAN KRIPTOGRAFI DENGAN ALGORITMA DES *(DATA ENCRYPTION STANDARD)* DAN BLOWFISH PADA SMS BERBASIS ANDROID

Nama : Antok Tega W
Pekerjaan : Mahasiswa
OS Android : jelly bean

No.	Pertanyaan	Jawaban	
		Ya	Tidak
1	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish dapat berjalan pada versi android Anda?	√	
2	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish tampilannya menarik?	√	
3	Apakah aplikasi SMS kriptografi ini mudah dioperasikan?		√
4	Apakah aplikasi SMS kriptografi ini bermanfaat untuk mengamankan pesan?	√	
5	Apakah proses enkripsi pesan berhasil pada aplikasi SMS kriptografi ini?	√	
6	Apakah proses dekripsi pesan berhasil pada aplikasi SMS kriptografi ini?	√	

Catatan: jawablah masing-masing pertanyaan dengan memberikan tanda centang (√) pada salah satu jawaban Ya atau Tidak.

Pengguna

(.....)

KUISIONER RANCANG BANGUN PENERAPAN KRIPTOGRAFI DENGAN ALGORITMA DES *(DATA ENCRYPTION STANDARD)* DAN BLOWFISH PADA SMS BERBASIS ANDROID

Nama : Priya Hamzah A

Pekerjaan : Mahasiswa

OS Android : Marsmallow

No.	Pertanyaan	Jawaban	
		Ya	Tidak
1	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish dapat berjalan pada versi android Anda?	✓	
2	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish tampilannya menarik?		✓
3	Apakah aplikasi SMS kriptografi ini mudah dioperasikan?		✓
4	Apakah aplikasi SMS kriptografi ini bermanfaat untuk mengamankan pesan?	✓	
5	Apakah proses enkripsi pesan berhasil pada aplikasi SMS kriptografi ini?	✓	
6	Apakah proses dekripsi pesan berhasil pada aplikasi SMS kriptografi ini?	✓	

Catatan: jawablah masing-masing pertanyaan dengan memberikan tanda centang (√) pada salah satu jawaban Ya atau Tidak.

Pengguna

(Priya. Hamzah. A)

KUISIONER RANCANG BANGUN PENERAPAN KRIPTOGRAFI DENGAN ALGORITMA DES *(DATA ENCRYPTION STANDARD)* DAN BLOWFISH PADA SMS BERBASIS ANDROID

Nama : Fitri kurniawati

Pekerjaan : Mahasiswa

OS Android : Samsung Galaxy core II (Lollipop)

No.	Pertanyaan	Jawaban	
		Ya	Tidak
1	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish dapat berjalan pada versi android Anda?	✓	
2	Apakah aplikasi SMS kriptografi dengan algoritma DES dan Blowfish tampilannya menarik?	✓	
3	Apakah aplikasi SMS kriptografi ini mudah dioperasikan?	✓	
4	Apakah aplikasi SMS kriptografi ini bermanfaat untuk mengamankan pesan?	✓	
5	Apakah proses enkripsi pesan berhasil pada aplikasi SMS kriptografi ini?	✓	
6	Apakah proses dekripsi pesan berhasil pada aplikasi SMS kriptografi ini?	✓	

Catatan: jawablah masing-masing pertanyaan dengan memberikan tanda centang (√) pada salah satu jawaban Ya atau Tidak.

Pengguna

(Fitri Kurniawati)

Source Code

1. sms_kripto.java

```
package com.android.danang_laptop.danang_178_sms_kriptografi;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ImageButton;
import android.widget.TextView;


public class sms_kripto extends Activity {

   public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.content_sms_kripto);
        final TextView textView1 = (TextView)
findViewById(R.id.textView1);
        textView1.setSelected(true);
        final ImageButton btnKirim = (ImageButton)
findViewById(R.id.btnKirim);
        final ImageButton btnBaca = (ImageButton) findViewById(R
.id.btnBaca);
        final ImageButton btnBantuan = (ImageButton)
findViewById(R.id.btnHelp);
        final ImageButton btnTentang = (ImageButton)
findViewById(R.id.btnTentang);
        final ImageButton btnKeluar = (ImageButton)
findViewById(R.id.btnKeluar);

       final Animation animAlpha =
AnimationUtils.loadAnimation(this, R.anim.anim_alpha);
       final Animation animScale =
AnimationUtils.loadAnimation(this, R.anim.anim_scale);

        btnKirim.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                arg0.startAnimation(animScale);
                Intent intent = new Intent(getBaseContext(),
send_sms.class);
                startActivity(intent);


            }
        });

        btnBaca.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View arg0) {
```

```
                // TODO Auto-generated method stub
                arg0.startAnimation(animScale);
                Intent intent = new Intent(getBaseContext(),
inbox_sms.class);
                startActivity(intent);


            }
        });
        btnBantuan.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                arg0.startAnimation(animAlpha);
                Intent intent = new Intent(getBaseContext(),
bantuan.class);
                startActivity(intent);
            }
        });
        btnTentang.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                arg0.startAnimation(animAlpha);
                Intent intent = new Intent(getBaseContext(),
tentang.class);
                startActivity(intent);
            }
        });

        btnKeluar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View arg0) {
                arg0.startAnimation(animAlpha);
                AlertDialog keluar = new
AlertDialog.Builder(sms_kripto.this).create();
                keluar.setMessage("Apakah anda ingin keluar?");
                keluar.setButton(-1, "Ya", new
DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface cobatok,
int arg1) {
                        sms_kripto.this.finish();
                    }
                });
                keluar.setButton(-2,"Tidak", new
DialogInterface.OnClickListener(){
                    public void onClick(DialogInterface cobatok,
int arg1){
                        cobatok.cancel();
                    }
                });
                keluar.show();
            }
        });

    }
}
```

2. send_sms.java

```
package com.android.danang_laptop.danang_178_sms_kriptografi;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.app.Activity;
import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.view.View;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class send_sms extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.content_send_sms);
        final Button btnEnkripsiDES = (Button)
findViewById(R.id.btnEnk);
        final Button btnGabungan = (Button)
findViewById(R.id.btnGab);
        final EditText edtPhone = (EditText)
findViewById(R.id.edtNoTujuan);
        final EditText edtPesan =(EditText)
findViewById(R.id.edtPesan);
        final Button btnKirim = (Button)
findViewById(R.id.btnSend);
        final TextView txtDES = (TextView)
findViewById(R.id.txtDES);
        final TextView txtBlow = (TextView)
findViewById(R.id.txtBlow);
        final TextView txtGab = (TextView)
findViewById(R.id.txtGab);
        final EditText edtKey= (EditText)
findViewById(R.id.edtKey);
        final EditText edtKey2= (EditText)
findViewById(R.id.edtKey2);

        final Animation animAlpha =
AnimationUtils.loadAnimation(this, R.anim.anim_alpha);

        btnEnkripsiDES.setOnClickListener(new
View.OnClickListener() {

            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                // arg0.startAnimation(animAlpha);
```

```
                String pesan = edtPesan.getText().toString();
                String strKey=edtKey.getText().toString().trim();
                String chiperTextDES= KriptoDES.enkrip(pesan,
strKey);
                if(strKey.length()==8){
                    txtDES.setText(chiperTextDES);

                }else {
                    AlertDialog salah = new
AlertDialog.Builder(send_sms.this).create();
                    salah.setTitle("Error Message !");
                    salah.setMessage("Key Atau Kunci DES Harus 8
Karakter !");
                    salah.setButton("OK", new
DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface coba,
int arg1) {
                            coba.cancel();
                        }
                    });
                    salah.show();
                }

            }
        });
        btnGabungan.setOnClickListener(new View.OnClickListener() {

            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                //arg0.startAnimation(animAlpha);
                String pesanchip = txtDES.getText().toString();
                String strKey =
edtKey2.getText().toString().trim();
                String chiperText =
KriptoBlowfish.enkripsigab(pesanchip, strKey);

                if (strKey.length() > 0) {
                    txtGab.setText(chiperText);

                } else {
                    AlertDialog salah = new
AlertDialog.Builder(send_sms.this).create();
                    salah.setTitle("Error Message !");
                    salah.setMessage("Harap isi Key atau Kunci
untuk Enkripsi !");
                    salah.setButton("OK", new
DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface coba,
int arg1) {
                            coba.cancel();
                        }
                    });
                    salah.show();
                }
            }
        });

        btnKirim.setOnClickListener(new View.OnClickListener() {
```

```
            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                //arg0.startAnimation(animAlpha);
                String noHP= edtPhone.getText().toString();
                String pesanchip = txtDES.getText().toString();
                String strKey=edtKey2.getText().toString().trim();
                String chiperText=
KriptoBlowfish.enkripsigab(pesanchip, strKey);
                if(noHP.length()>0){
                    kirimSMS(noHP, chiperText);
                    txtGab.setText(chiperText);
                }else {
                    AlertDialog salah = new
AlertDialog.Builder(send_sms.this).create();
                    salah.setTitle("Error Message !");
                    salah.setMessage("Nomor Telepon Penerima Pesan
Masih Kosong !");
                    salah.setButton("OK", new
DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface coba,
int arg1) {
                            coba.cancel();
                        }
                    });
                    salah.show();
                }
            }
        });


    }

    public void kirimSMS(String noHP, String pesan){
        String SENT = "SMS_SENT";
        String DELIVERED = "SMS_DELIVERED";
        PendingIntent sentPI = PendingIntent.getBroadcast(this, 0,
new Intent(SENT), 0);
        PendingIntent deliveredPI =
PendingIntent.getBroadcast(this, 0, new Intent(DELIVERED), 0);
        //ketika SMS SENT
        registerReceiver(new BroadcastReceiver() {
            @Override
            public void onReceive(Context context, Intent intent) {
                // TODO Auto-generated method stub
                switch (getResultCode()) {
                    case Activity.RESULT_OK:
                        Toast.makeText(getBaseContext(), "SMS
SENT", Toast.LENGTH_LONG).show();

                        break;
                    case
android.telephony.SmsManager.RESULT_ERROR_GENERIC_FAILURE:
                        Toast.makeText(getBaseContext(), "ERROR
BOZZ!!", Toast.LENGTH_LONG).show();
                        break;
                    case
android.telephony.SmsManager.RESULT_ERROR_NO_SERVICE:
```

```
                        Toast.makeText(getBaseContext(), "TAK ADA
SIGNAL", Toast.LENGTH_LONG).show();
                        break;
                    case
android.telephony.SmsManager.RESULT_ERROR_NULL_PDU:
                        Toast.makeText(getBaseContext(), "PDU
NULL", Toast.LENGTH_LONG).show();
                        break;
                    case
android.telephony.SmsManager.RESULT_ERROR_RADIO_OFF:
                        Toast.makeText(getBaseContext(), "ERROR
BOZZ, GSM MATI", Toast.LENGTH_LONG).show();
                        break;

                    default:
                        Toast.makeText(getBaseContext(), "Wah ga
tau dah ni..", Toast.LENGTH_LONG).show();
                }
            }
        }, new IntentFilter(SENT));

        //ketika SMS DELIVERED
        registerReceiver(new BroadcastReceiver() {
            @Override
            public void onReceive(Context context, Intent intent) {
                // TODO Auto-generated method stub
                switch (getResultCode()) {
                    case Activity.RESULT_OK:
                        Toast.makeText(getBaseContext(), "SMS
TERKIRIM BOZZ :-)", Toast.LENGTH_LONG).show();

                        break;
                    case Activity.RESULT_CANCELED:
                        Toast.makeText(getBaseContext(), "SMS TIDAK
TERKIRIM BOZZ!!", Toast.LENGTH_LONG).show();
                        break;


                    default:
                        Toast.makeText(getBaseContext(), "??",
Toast.LENGTH_LONG).show();
                }

            }

        }, new IntentFilter(DELIVERED));

        android.telephony.SmsManager sms =
android.telephony.SmsManager.getDefault();
        sms.sendTextMessage(noHP, null, pesan, sentPI,
deliveredPI);

    }


}
```

3. inbox sms.java

```
package com.android.danang_laptop.danang_178_sms_kriptografi;

import java.util.ArrayList;
import java.util.List;

import android.net.Uri;
import android.os.Bundle;
import android.app.Activity;
import android.database.Cursor;
import android.view.View;
import android.widget.AcapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.TextView;


public class inbox_sms extends Activity {
      @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.content_inbox_sms);

          final ListView list = (ListView)
findViewById(R.id.list);
          final Button btnKey = (Button)
findViewById(R.id.btnKey);
          final Button btnKey2 = (Button)
findViewById(R.id.btnKey2);
          final TextView txtChipdes =
(TextView)findViewById(R.id.txtChipdes);



        btnKey.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                final EditText edtKey = (EditText)
findViewById(R.id.edtKey);
                List<String> msgs =
getSMS(edtKey.getText().toString().trim());
                if (msgs.isEmpty()) {
                    msgs.add("Tidak ada SMS yang bisa dipecahkan
dengan key tsbt");
                }
                ArrayAdapter<String> smsAdapter = new
ArrayAdapter<String>(getBaseContext(),
                        android.R.layout.simple_list_item_1,
msgs);
                list.setAdapter(smsAdapter);


            }

        });
```

```
            list.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
                public void onItemClick(AdapterView<?> parent,
View view, int position, long id) {
                    String item = ((TextView)
view).getText().toString();
                    txtChipdes.setText(item);
                }

            });

        btnKey2.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                final EditText edtKey2 = (EditText)
findViewById(R.id.edtKey2);
                final TextView mesg =
getDES(edtKey2.getText().toString().trim());
                if (mesg.length() == 0) {
                    mesg.setText("Tidak ada SMS yang bisa
dipecahkan dengan key tsbt");
                }

            }
        });

    }


    public List<String>  getSMS(String strkey) {
        List<String> list = new ArrayList<String>();
        Uri uri = Uri.parse("content://sms/inbox");
        Cursor c = null;
        try{
            c =
getApplicationContext().getContentResolver().query(uri, null,
null ,null,null);
        }catch(Exception e){
            e.printStackTrace();
        }
        try{
            for (boolean hasData = c.moveToFirst(); hasData;
hasData = c.moveToNext()) {

                final String msg =
c.getString(c.getColumnIndexOrThrow("body"));

                String plainTeks= KriptoBlowfish.dekripsi(msg,
strkey);
                if(!plainTeks.equalsIgnoreCase("ERROR")){
                    list.add(plainTeks);

                }

            }
        }catch(Exception e){
            e.printStackTrace();
```

```
        }
        c.close();
        return list;
    }

       public TextView getDES(String strkey) {

           final TextView txtChipdes =
(TextView)findViewById(R.id.txtChipdes);
           final TextView listAsli = (TextView)
findViewById(R.id.listAsli);

        Uri uri = Uri.parse("content://sms/inbox");
        Cursor k = null;

        try{
            k =
getApplicationContext().getContentResolver().query(uri, null,
null ,null,null);
        }catch(Exception e){
            e.printStackTrace();
        }
        try{
            for (boolean hasData = k.moveToFirst(); hasData;
hasData = k.moveToNext()) {

                final String noHP =
k.getString(k.getColumnIndex("address"));

                String chiperDES =
txtChipdes.getText().toString();

                String teksAsli= KriptoDES.dekrip(chiperDES,
strkey);
                if(!teksAsli.equalsIgnoreCase("ERROR")){
                    listAsli.setText("SMS dari: " + noHP +
"\nText: " + teksAsli);

                }

            }
        }catch(Exception e){
            e.printStackTrace();
        }
        k.close();
        return listAsli;
    }


}
```

4. ReceiveSMS.java

```
package com.android.danang_laptop.danang_178_sms_kriptografi;

/**
 * Created by DANANG-LAPTOP on 10/04/2016.
 */
import android.content.BroadcastReceiver;

import android.content.Context;

import android.content.Intent;

import android.os.Bundle;

//import android.telephony.gsm.SmsMessage;

import android.telephony.SmsMessage;
import android.widget.Toast;


public class ReceiveSMS extends BroadcastReceiver

{

    @Override

    public void onReceive(Context context, Intent intent)

    {

        Bundle bundle = intent.getExtras();

        SmsMessage[] msgs = null;

        String str = "";

        if (bundle != null)

        {

            Object[] pdus = (Object[]) bundle.get("pdus");

            msgs = new SmsMessage[pdus.length];
            String strMsg="";

            for (int i=0; i<msgs.length; i++){

                msgs[i] =
SmsMessage.createFromPdu((byte[])pdus[i]);

                str += "SMS dari " +
msgs[i].getOriginatingAddress();
                str += " :";

                str += msgs[i].getMessageBody().toString();

                str += "\n";
```

```
                strMsg=msgs[i].getMessageBody().toString();

            }
            Toast.makeText(context, "SMS MASUK:"+strMsg,
Toast.LENGTH_LONG).show();

            }

    }

}
```

5. KriptoDES.java

```
package com.android.danang_laptop.danang_178_sms_kriptografi;

/**
 * Created by DANANG-LAPTOP on 18/04/2016.
 */

import android.util.Base64;

import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

public class KriptoDES {

    public static String enkrip(String pesan, String key){
        try {
            SecretKeySpec SK = new SecretKeySpec(key.getBytes(),
"DES");
            Cipher cipher = Cipher.getInstance("DES");
            cipher.init(Cipher.ENCRYPT_MODE, SK);
           byte[] encrypted = cipher.doFinal(pesan.getBytes());
            return Base64.encodeToString(encrypted,
Base64.NO_PADDING);
        } catch (Exception e) {
            return "ERROR:" + e.getMessage();
        }
    }

   public static String dekrip(String txtChipdes, String key){
        try {
            SecretKeySpec SK = new SecretKeySpec(key.getBytes(),
"DES");
            Cipher cipher = Cipher.getInstance("DES");
            cipher.init(Cipher.DECRYPT_MODE, SK);
            byte[] decrypted =
cipher.doFinal(Base64.decode(txtChipdes, Base64.NO_PADDING));
            return new String(decrypted);
        } catch (Exception e) {
            return "ERROR";
        }
    }
}
```

6. KriptoBlowfish.java

```
package com.android.danang_laptop.danang_170_sms_kriptografi;

import android.util.Base64;

import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;

/**
 * Created by DANANG-LAPTOP on 19/04/2016.
 */
public class KriptoBlowfish {
    public static String enkripsi(String pesan, String key){
        try {
            SecretKeySpec KS = new SecretKeySpec(key.getBytes(),
"Blowfish");
            Cipher cipher = Cipher.getInstance("Blowfish");
            cipher.init(Cipher.ENCRYPT_MODE, KS);
            byte[] encrypted = cipher.doFinal(pesan.getBytes());
            return
Base64.encodeToString(encrypted,Base64.NO_PADDING);
        } catch (Exception e) {
            return "ERROR:"+e.getMessage();
        }
    }

    public static String enkripsigab (String pesanchip, String
key){
        try {
            SecretKeySpec KS = new SecretKeySpec(key.getBytes(),
"Blowfish");
            Cipher cipher = Cipher.getInstance("Blowfish");
            cipher.init(Cipher.ENCRYPT_MODE, KS);
            byte[] encrypted =
cipher.doFinal(pesanchip.getBytes());
            return Base64.encodeToString(encrypted,
Base64.NO_PADDING);
        } catch (Exception e) {
            return "ERROR:"+e.getMessage();
        }
    }

    public static String dekripsi(String txtGab, String key){
        try {

            SecretKeySpec KS = new SecretKeySpec(key.getBytes(),
"Blowfish");
            Cipher cipher = Cipher.getInstance("Blowfish");
            cipher.init(Cipher.DECRYPT_MODE, KS);
            byte[] decrypted =
cipher.doFinal(Base64.decode(txtGab, Base64.NO_PADDING));
            return new String(decrypted);
        } catch (Exception e) {
            return "ERROR";
        }
    }
}
```

7. content_sms_kripto.xml

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/LinearLayout1"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/encl"
    android:orientation="vertical"
    android:weightSum="1">


    <TextView
        android:id="@+id/textView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="SMS Kriptografi Dengan Algoritma DES dan
Blowfish"
        android:ellipsize="marquee"
        android:marqueeRepeatLimit="marquee_forever"
        android:singleLine="true"

android:textAppearance="?android:attr/textAppearanceMedium"
        android:textColor="#a8f8e4"
        android:textAlignment="center"
        android:textStyle="bold"
        android:layout_gravity="center_horizontal"
        android:textIsSelectable="false"
        android:background="#031c6f"
        android:layout_weight="0.2" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text=""
        android:id="@+id/textView14"
        android:layout_gravity="center_horizontal" />

    <ImageButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/btnKirim"
        android:layout_gravity="center_horizontal"
        android:src="@drawable/tulissms"
        style="@android:style/Holo.ButtonBar" />

    <ImageButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/btnBaca"
        android:src="@drawable/bacasms"
        style="@android:style/Holo.ButtonBar"
        android:layout_gravity="center_horizontal"
        android:layout_weight="0.29" />


    <LinearLayout
```

```
        android:orientation="horizontal"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal">

        <ImageButton
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/btnHelp"
            android:layout_gravity="center_horizontal"
            android:src="@drawable/help"
            style="@android:style/Holo.ButtonBar" />

        <TextView
            android:layout_width="19dp"
            android:layout_height="match_parent"
            android:text=""
            android:id="@+id/textView47" />

        <ImageButton
            android:id="@+id/btnTentang"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:src="@drawable/tentang"
            style="@android:style/Holo.ButtonBar" />
    </LinearLayout>

    <ImageButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/btnKeluar"
        android:layout_gravity="center"
        android:src="@drawable/exit"
        style="@android:style/Holo.ButtonBar" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="New Text"
        android:id="@+id/textView48"
        android:layout_gravity="center_horizontal"
        android:layout_weight="0.2" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="dhanank4xbar@gmail.com | T.Informatika -
ITN Malang"
android:textAppearance="?android:attr/textAppearanceMedium"
        android:textColor="#a8f8e4"
        android:textStyle="bold|italic" />

</LinearLayout>
```

8. content_send_sms.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/LinearLayout1"

    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:baselineAligned="false"
    android:nestedScrollingEnabled="false"
    android:orientation="vertical">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="No Tujuan" />


    <EditText
        android:id="@+id/edtNoTujuan"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:phoneNumber="true">

        <requestFocus />
    </EditText>

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Pesan" />

    <EditText
        android:id="@+id/edtPesan"
        android:layout_width="match_parent"
        android:layout_height="140dp"
        android:ems="10"
        android:inputType="textMultiLine" />


    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Key DES (Panjang Key 8 Karakter)"

android:textAppearance="?android:attr/textAppearanceSmall" />

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content">

        <EditText
            android:id="@+id/edtKey"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:ems="10" />

        <Button
            android:id="@+id/btnEnk"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:text="Enkripsi DES"
            android:layout_gravity="center_horizontal"

android:textAppearance="?android:attr/textAppearanceSmall"

android:background="@android:drawable/btn_default_small"
            android:textStyle="bold|normal" />

    </LinearLayout>


    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Key Blowfish"

android:textAppearance="?android:attr/textAppearanceSmall" />

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:weightSum="1">

        <EditText
            android:id="@+id/edtKey2"
            android:layout_width="191dp"
            android:layout_height="wrap_content"
            android:ems="10" />

        <Button
            android:id="@+id/btnGab"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Enkripsi DES + Blowfish"

android:textAppearance="?android:attr/textAppearanceSmall"
            android:letterSpacing="0"

android:background="@android:drawable/btn_default_small"
            android:textStyle="bold|normal" />

    </LinearLayout>

    <TextView
```

```
        android:layout_width="match_parent"
        android:layout_height="15dp"
        android:id="@+id/textView27" />

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:weightSum="1">

        <Button
            android:id="@+id/btnSend"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="KIRIM SMS"
            android:background="@android:drawable/btn_default"

android:textAppearance="?android:attr/textAppearanceSmall"
            android:allowUndo="false"
            android:textStyle="bold|normal" />

    </LinearLayout>

    <TextView
        android:id="@+id/textView15"
        android:layout_width="match_parent"
        android:layout_height="5dp"

android:textAppearance="?android:attr/textAppearanceSmall" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:text="Hasil Enkripsi: "

android:textAppearance="?android:attr/textAppearanceSmall" />

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_weight="3.10"
        android:weightSum="1"
        android:baselineAligned="true"
        android:clickable="false" >

        <TextView
            android:id="@+id/txtGab"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
            android:nestedScrollingEnabled="false"
            android:layout_gravity="center_horizontal"
            android:layout_weight="15.23" />
    </LinearLayout>
```

```
    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_weight="3.10"
        android:weightSum="1"
        android:baselineAligned="true"
        android:clickable="false">

        <TextView
            android:id="@+id/txtDES"
            android:layout_width="104dp"
            android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
            android:allowUndo="false"
            android:nestedScrollingEnabled="false"
            android:textColor="#ff48b9"
            android:visibility="invisible" />

        <TextView
            android:id="@+id/txtBlow"
            android:layout_width="110dp"
            android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceMedium"
            android:nestedScrollingEnabled="false"
            android:textColor="#b52a2a"
            android:visibility="invisible" />

    </LinearLayout>

</LinearLayout>
</ScrollView>
```

9. content_inbox_sms.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/LinearLayout2"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="1">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="List SMS Kriptografi"
```

```
android:textAppearance="?android:attr/textAppearanceLarge" />


    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/textView5"
        android:layout_gravity="center_horizontal"
        android:text="Key Blowfish :" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:weightSum="1">


        <EditText
            android:id="@+id/edtKey"
            android:layout_width="266dp"
            android:layout_height="wrap_content"
            android:ems="10" />
        <Button
            android:id="@+id/btnKey"
            style="?android:attr/buttonStyleSmall"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="OK"
            android:textStyle="bold"

android:background="@android:drawable/btn_default_small" />

    </LinearLayout>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/textView8"
        android:layout_gravity="center_horizontal"
        android:text="Pilih Chiper DES :" />

    <ListView
        android:id="@+id/list"
        android:layout_width="match_parent"
        android:layout_height="130dp"
        android:layout_weight="0.38">
    </ListView>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/textView7"
        android:layout_gravity="center_horizontal"
        android:text="Chiper DES :" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="37dp"
        android:id="@+id/txtChipdes"
```

```
        android:layout_gravity="center_horizontal"
        android:textSize="15dp"
        android:textIsSelectable="true"
        android:textColor="#030000" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/textView9"
        android:layout_gravity="center_horizontal"
        android:text="Key DES (Masukan 8 Karakter) :" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:weightSum="1">

        <EditText
            android:id="@+id/edtKey2"
            android:layout_width="252dp"
            android:layout_height="wrap_content"
            android:ems="10" />

        <Button
            style="?android:attr/buttonStyleSmall"
            android:id="@+id/btnKey2"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="OK"
            android:textStyle="bold"

android:background="@android:drawable/btn_default_small" />
    </LinearLayout>

    <TextView
        android:id="@+id/textView6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Pesan Asli:"

android:textAppearance="?android:attr/textAppearanceMedium"
        android:textSize="15dp" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="66dp"
        android:id="@+id/listAsli"
        android:layout_gravity="center_horizontal"
        android:textColor="#050000"
        android:textSize="15dp" />

</LinearLayout>
</ScrollView>
```

10. content_bantuan.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/LinearLayout2"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="1">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="List SMS Kriptografi"

android:textAppearance="?android:attr/textAppearanceLarge" />


    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/textView5"
        android:layout_gravity="center_horizontal"
        android:text="Key Blowfish :" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:weightSum="1">


        <EditText
            android:id="@+id/edtKey"
            android:layout_width="266dp"
            android:layout_height="wrap_content"
            android:ems="10" />
        <Button
            android:id="@+id/btnKey"
            style="?android:attr/buttonStyleSmall"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="OK"
            android:textStyle="bold"

android:background="@android:drawable/btn_default_small" />

    </LinearLayout>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/textView8"
```

```
        android:layout_gravity="center_horizontal"
        android:text="Pilih Chiper DES :" />

    <ListView
        android:id="@+id/list"
        android:layout_width="match_parent"
        android:layout_height="130dp"
        android:layout_weight="0.38">
    </ListView>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/textView7"
        android:layout_gravity="center_horizontal"
        android:text="Chiper DES :" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="37dp"
        android:id="@+id/txtChipdes"
        android:layout_gravity="center_horizontal"
        android:textSize="15dp"
        android:textIsSelectable="true"
        android:textColor="#030000" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/textView9"
        android:layout_gravity="center_horizontal"
        android:text="Key DES (Masukan 8 Karakter) :" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:weightSum="1">

        <EditText
            android:id="@+id/edtKey2"
            android:layout_width="252dp"
            android:layout_height="wrap_content"
            android:ems="10" />

        <Button
            style="?android:attr/buttonStyleSmall"
            android:id="@+id/btnKey2"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="OK"
            android:textStyle="bold"

android:background="@android:drawable/btn_default_small" />
    </LinearLayout>

    <TextView
        android:id="@+id/textView6"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:text="Pesan Asli:"

android:textAppearance="?android:attr/textAppearanceMedium"
        android:textSize="15dp" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="66dp"
        android:id="@+id/listAsli"
        android:layout_gravity="center_horizontal"
        android:textColor="#050000"
        android:textSize="15dp" />

</LinearLayout>
</ScrollView>
```

11. content_tentang.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"

tools:context="com.android.danang_laptop.danang_178_sms_kriptogr
afi.tentang">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Tentang Informasi Aplikasi dan Pembuat
Aplikasi"
        android:id="@+id/textView28"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:gravity="center_horizontal"
        android:textStyle="bold"
        android:textSize="14dp" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Aplikasi SMS Kriptografi ini dibuat
sebagai aplikasi untuk pengamanan pesan SMS dari penyadap yang
tidak bertanggungjawab. Dengan Aplilasi ini dapat menjaga
rahasia pesan anda agar tidak tersebar ke orang yang tidak
bertanggung jawab."
        android:id="@+id/textView42"
```

```
        android:gravity="center_horizontal"
        android:layout_marginTop="27dp"
        android:layout_below="@+id/textView28"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true" />

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/imageView"
        android:src="@drawable/profilku"
        android:layout_centerVertical="true"
        android:layout_alignRight="@+id/textView46"
        android:layout_alignEnd="@+id/textView46" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Institut Teknologi Nasional Malang"
        android:id="@+id/textView43"
        android:gravity="center_horizontal"
        android:textStyle="bold"
        android:layout_below="@+id/textView44"
        android:layout_centerHorizontal="true" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Teknik Informatika S-1"
        android:id="@+id/textView44"
        android:gravity="center_horizontal"
        android:textStyle="bold"
        android:layout_below="@+id/textView45"
        android:layout_centerHorizontal="true" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="NGAWI, 25 September 1993"
        android:id="@+id/textView45"
        android:gravity="center_horizontal"
        android:textStyle="bold"
        android:layout_below="@+id/textView46"
        android:layout_centerHorizontal="true" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="DANANG NUR FADLILAH AKBAR"
        android:id="@+id/textView46"
        android:gravity="center_horizontal"
        android:textStyle="bold"
        android:layout_marginTop="25dp"
        android:layout_below="@+id/imageView"
        android:layout_centerHorizontal="true" />

</RelativeLayout>
```