SKRIPSI

DESAIN DAN IMPLEMENTASI APLIKASI *MOBILE* UNTUK ENKRIPSI SMS DENGAN ALGORITMA RC6

Disusun Oleh :

DELANIGA FAJARIA RUSLANI
NIM 06.12.584

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
AGUSTUS 2010

LEMBAR PERSETUJUAN

DESAIN DAN IMPLEMENTASI APLIKASI *MOBILE* UNTUK ENKRIPSI SMS DENGAN MENGGUNAKAN METODE ALGORITMA RC6

SKRIPSI

Disusun dan Diajukan sebagai Salah Satu Syarat Untuk Memperoleh

Gelar Sarjana Teknik Komputer Dan Informatika Strata Satu (S–1)

Disusun Oleh :

DELANIGA FAJARIA RUSLANI

NIM : 06.12.684

Diperiksa dan Disetujui

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Yusuf Ismail Nakhoda, MT	**Achmad Faisol, ST**
NIP Y. 1018800189	

Mengetahui

Ketua Jurusan Teknik Elektro S-1

Ir. Yusuf Ismail Nakhoda, MT

NIP Y. 1018800189

JURUSAN TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2010

Desain dan Implementasi Aplikasi *Mobile* Untuk Enkripsi SMS Dengan Algoritma RC6

Delaniga Fajaria Ruslani

Jurusan Teknik Elektro S-1, Konsentrasi T.Komputer dan Informatika
Fakultas Teknologi Industri, Institut Teknologi Nasional Malang
Jln. Raya Karanglo Km 2 Malang
d3la_88@yahoo.co.id

Dosen Pembimbing : I. Ir. Yusuf Ismail Nakhoda, MT.
II. Ahmad Faisol, ST.

Abstraksi

Masalah keamanan dan kerahasiaan merupakan salah satu aspek penting dari suatu pesan, data, atau informasi. Dalam hal ini sangat terkait dengan pentingnya pesan, data, atau informasi tersebut dikirim dan diterima oleh pihak atau orang yang berkepentingan, apakah pesan, data, atau informasi masih asli atau terdapat perubahan oleh pihak lain. Pesan, data, atau informasi akan tidak berguna lagi apabila di tengah jalan informasi itu disadap atau dibajak oleh orang yang tidak berhak atau berkepentingan. Salah satu komunikasi saat ini adalah *Short Messaging Services* (SMS) merupakan cara untuk berkomunikasi yang populer saat ini. Dengan tarif yang murah, seseorang dapat bertukar informasi dengan orang lain. Alasan lain orang memilih SMS karena pengirimannya yang cepat. Namun terdapat kelemahan pada jaringan pengiriman SMS tersebut yang dapat dimanfaatkan untuk melakukan *Spoofing*.

Dengan adanya masalah tersebut untuk melindungi informasi atau isi pesan agar tidak terlihat oleh orang atau pihak yang tidak berhak, maka dapat cara melakukan enkripsi pada informasi atau isi pesan agar menjadi samar. Algoritma RC6 adalah salah satu algoritma kriptografi jenis Block chipper yang dapat dimafaatkan untuk melakukan enkripsi isi pesan tersebut. Oleh karena itu dibuat aplikasi SMS RC6 berbasis JAVA (*J2ME*) yang mengamankan isi SMS dengan melakukan enkripsi pada saat pengiriman SMS, dan penerima SMS akan melakukan dekripsi pada saat membuka SMS

Kata kunci : *SMS, aplikasi, kriptografi, RC6, Spoofing*

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa yang telah memberikan rahmat dan hidayah-Nya, sehingga dapat diselesaikan skripsi yang berjudul **"DESAIN DAN IMPLEMENTASI APLIKASI *MOBILE* UNTUK ENKRIPSI SMS DENGAN MENGGUNAKAN ALGORITMA RC6"** ini dengan lancar. Skripsi ini merupakan persyaratan kelulusan Studi pada Jurusan Teknik Elektro S-1 Konsentrasi Teknik Komputer dan Informatika ITN Malang dan untuk mencapai gelar Sarjana Teknik.

Keberhasilan peyelesaian laporan skripsi ini tidak lepas dari dukungan dan bantuan berbagai pihak. Untuk itu penyusun menyampaikan terima kasih kepada :

1. Prof. Dr. Ir. Abraham Lomi, MSEE, selaku Rektor Institut Teknologi Nasional Malang.
2. Bapak Ir. Sidik Noertjahjono, MT, selaku Dekan Fakultas Teknologi Industri.
3. Bapak Ir. F. Yudi Limpraptono, MT selaku Ketua Jurusan Teknik Elektro S-1.
4. Bapak Ir. Yusuf Ismail Nakhoda, MT selaku Sekretaris Jurusan Teknik Elektro S-1.
5. Bapak Ir. Yusuf Ismail Nakhoda, MT selaku Dosen Pembimbing I.
6. Bapak Ahmad Faisol, ST. selaku Dosen Pembimbing II.
7. Ibu serta saudara-saudara kami yang telah memberikan do'a restu, dorongan, semangat, dan biaya.

8. Rekan-rekan instruktur di Laboratorium Pemrograman dan Multimedia ITN Malang.
9. Semua yang telah membantu dalam penyelesaian penyusunan skripsi ini.

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

Harapan penyusun semoga laporan skripsi ini memberikan manfaat bagi perkembangan ilmu pengetahuan dan pembaca.

Malang, Agustus 2010

penyusun

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG

Telepon seluler merupakan alat komunikasi yang sudah dipakai oleh sebagian besar orang di dunia. Telepon seluler menyediakan media komunikasi yang beragam dan salah satu cara diantaranya adalah media SMS *(Short Message Service)*. SMS merupakan suatu layanan yang memungkinkan pengguna telepon seluler untuk mengirimkan pesan singkat kepada pengguna telepon seluler lainnya. SMS bekerja pada sistem nirkabel, sistem nirkabel yang paling popular di dunia adalah GSM *(Global System for Mobile Communication)*. Layanan pengiriman pesan singkat sangatlah standard dan tidak jarang para pengguna telepon seluler menggunakan layanan SMS ini untuk mengirimkan suatu pesan yang penting dan rahasia, namun para pengguna layanan SMS tersebut sering kali tidak mengetahui bahwa jalur komunikasi SMS memiliki banyak sekali celah yang memungkinkan untuk terjadinya serangan pada pesan teks yang dikirim.

Alasan dari penggunaan SMS adalah tarifnya yang murah, praktis dan pengirimannya yang cepat. Tetapi dengan maraknya penyadapan SMS oleh berbagai pihak juga adanya *Spoofing* maka tingkat keamanan SMS berkurang. SMS *Spoofing* adalah pengiriman SMS dimana nomor pengirim yang tertera bukanlah nomor pengirim yang sebenarnya. Mekanisme SMS *Spoofing* ini dimungkinkan karena lemahnya proteksi koneksi SMS *Service Centre* (SMSC) *gateway*. Penyusup dapat merekam *login* dan *password* dari pesan yang berasal dari SMS *gateway* menuju

SMSC. Walaupun tak terlalu mudah namun ini dapat dilakukan dalam beberapa kasus.

Dalam hal ini penyusup mengatur sebuah gateway palsu yang berlaku seperti *gateway* sesungguhnya. *Gateway* palsu ini dapat mengirim semua jenis pesan pendek kepada user *Mobile Station* (MS) melalui SMSC. Pada teknik *spoofing* ini pesan dikirim dengan memanipulasi nomor *Mobile Subscriber Integrated Services Digital Network Number* (MSISDN) asal (*originate*) pada *field* yang disediakan sehingga pesan akan tampak datang dari nomor pengirim lainnya.[1]

Keamanan dan kerahasiaan sebuah data atau informasi pesan SMS dalam komunikasi dan pertukaran informasi sangatlah penting. Seringkali data atau informasi yang penting, dalam komunikasi dan pertukaran informasi kadang tidak sampai kepada penerima atau tidak hanya diterima oleh penerima tetapi juga oleh pihak lain yang melakukan pembajakan atau penyadapan. Hal ini membuat data atau informasi tersebut menjadi tidak berguna lagi dan lebih parahnya lagi kadang data atau informasi tersebut oleh para pembajak digunakan untuk menjatuhkan pihak lain. Ini dikarenakan terdapat kemungkinan pihak lain dapat mencuri informasi yang disampaikan dalam SMS karena masih lemahnya keamanan dalam SMSC *gateway* yang dapat digunakan sebagai celah untuk melakukan SMS *Spoofing*, maka salah satu alternatif yang dapat digunakan untuk menjaga kerahasiaan informasi tersebut adalah dengan melakukan enkripsi dengan menggunakan algoritma pada pesan yang dikirimkan.

Oleh karena itu enkripsi dengan menggunakan algoritma sangat dibutuhkan dalam menjaga kerahasiaan data atau informasi. Algoritma kriptografi terdiri dari algoritma enkripsi (E) dan algoritma dekripsi (D). Enkripsi dimaksudkan untuk

melindungi informasi agar tidak terlihat oleh orang atau pihak yang tidak berhak. Ada banyak model dan metode enkripsi, salah satu di antaranya adalah enkripsi dengan algoritma *Rivest Code* 6 (RC6). Model ini merupakan salah satu algoritma kunci simetris yang berbentuk *block chiper* yang merupakan salah satu kandidat dalam penentuan algoritma standart untuk kriptografi AES *(Adnvanced Encryption Standard).* Maka sebuah aplikasi dirancang untuk dapat mengimplementasikan algoritma RC6. Algoritma RC6 yang dirancang oleh Ronald L. Rivest, M.J.B. Robshaw, R. Sidney, dan Y.L. Yin merupakan salah satu algoritma yang menjadi finalis kandidat untuk menjadi AES.

1.2 RUMUSAN MASALAH

Rumusan masalah yang dapat diambil dari pembuatan tugas akhir ini adalah :

Adapun rumusan masalah dalam penelitian ini adalah bagaimana cara merahasiakan pesan singkat atau SMS dari pihak-pihak yang tidak berhak mengetahui isi pesan tersebut. Serta mengimplementasikan algoritma RC6 pada telepon seluler untuk keamanan data dan informasi.

1.3 TUJUAN PENELITIAN

Tujuan yang akan dicapai dalam pelaksanan tugas akhir ini diantaranya :

Tujuan dari penelitian ini adalah memanfaatkan kriptografi untuk melindungi suatu informasi yang dikirim melalui pesan singkat atau SMS dari pihak-pihak yang tidak berhak mengetahui isi pesan singkat tersebut.

1.4 BATASAN MASALAH

Batasan masalah yang diambil pada penulisan skripsi ini diharapkan mampu membatasi pembahasan agar sesuai dengan tujuan penelitian itu sendiri. Adapun batasan masalah yang diajukan adalah sebagai berikut :

a) Aplikasi yang dibuat untuk pengiriman pesan antara dua *handphone*.

b) Aplikasi yang dibangun merupakan aplikasi yang berdiri sendiri, akan terpisah dari aplikasi SMS standar yang dimiliki oleh telepon selular yang menjadi sasaran implementasi.

c) Tidak membahas *Record Management System (RMS),* karena aplikasi yang dibuat tidak terdapat fitur penyimpanan pesan seperti inbox atau *outbox*.

d) Tidak membahas *basic* fitur SMS seperti *Message Submission and Delivery, Status Repot, Reply Path, Addressing Mode* dan *Validity Period.*

e) Tidak membahas arsitektur SMS seperti *Short Message Entity, SMS Service Center,* dan *Email Gateway.*

f) Aplikasi yang dibuat dapat diinstal hanya pada *handphone* yang mempunyai *Java Virtual Machine* dan minimal mempunyai *Mobile Information Device Profile (MIDP)* 2.0.

1.5 METODOLOGI

Adapun metodologi yang digunakan adalah sebagai berikut :

a) Studi Literatur

Mempelajari dan mengumpulkan literatur baik dari buku-buku, maupun internet yang berkaitan dengan judul sehingga dapat mendukung penyusunan skripsi ini.

b) Perancangan

Merancang dan mengembangkan desain aplikasi untuk proses enkripsi dan deskripsi SMS.

c) Pembuatan Program

Pembuatan Aplikasi dengan menggunakan bahasa pemrograman JAVA.

d) Implementasi dan Pengujian.

Merancang dan mengembangkan desain aplikasi SMS berdasarkan data-data yang diperoleh.

1.6 SISTEMATIKA PENULISAN

Untuk mempermudah dan memahami pembahasan penulisan skripsi ini, maka sistematika penulisan disusun sebagai berikut :

Bab I : Pendahuluan

Berisi penjelasan mengenai Latar Belakang, Rumusan Masalah, Tujuan Penelitian, Batasan Masalah, Metodologi, serta Sistematika Penulisan yang digunakan untuk menyusun Laporan Tugas Akhir.

Bab II : Landasan Teori

Berisi tentang landasan teori mengenai permasalahan yang berhubungan dengan pembahasan yang dilakukan, yang di dalamnya memuat teori-teori tentang SMS, Kriptografi dan Java

Bab III : Analisis Penyelesaian Masalah

Berisi analisis terhadap penggunaan algoritma RC6 dalam enskripsi dan deskripsi SMS pada telepon selular sehingga dapat membantu dalam melakukan perancangan dan implementasi.

Bab IV : Implementasi dan Pengujian

Bab ini berisi tentang Implementasi dari Hasil Desain Aplikasi pada emulator dan pengujian pada aplikasi

Bab VI : Penutup

Bab ini berisi tentang kesimpulan dan saran dari hasil penyusunan laporan Tugas Akhir yang telah disusun.

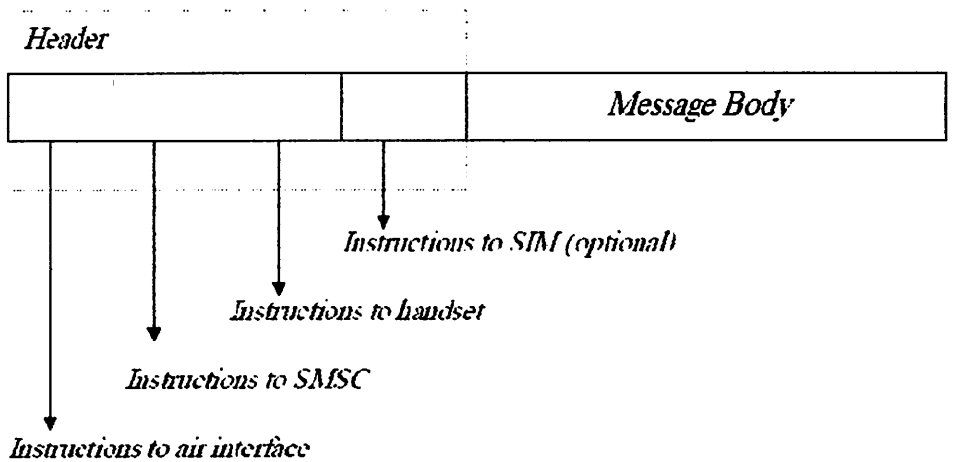
BAB II

TINJAUAN PUSTAKA

2.1 STRUKTUR PESAN SMS *(Short Message Service)*

Short Message Services (SMS) merupakan sebuah layanan yang banyak diaplikasikan pada system telekomunikasi tanpa kabel yang memungkinkan dilakukannya pengiriman pesan dalam bentuk *alphanumeric*. SMS dapat digunakan untuk mengirim pesan teks ke sesama pengguna *handphone*, SMS juga dapat digunakan untuk pengiriman email dan lain-lain.

Struktur pesan pada sebuah paket SMS dapat dilihat pada gambar berikut :



Gambar 1.2 Struktur Pesan SMS

Pada gambar dapat terlihat bahwa pada sebuah paket pesan SMS terdiri dari *header* dan *body. Header* pesan terdiri dari instrukturi-instruksi kepada komponen-komponen yang bekerja dalam jaringan SMS. Pada insturksi-instruksi tersebut terdapat informasi yang diperlukan selama pegiriman pesan seperti informasi validitas pesan, dan informasi-informasi lainnya. Pada bagian *message body,* terdapat isi dai pengirim pesan yang akan dikirimkan.

Panjang isi sebuah pesan paket SMS berukuran maksimal 160 karakter, dimana setiap karakter memiliki panjang 7 bit. Beberapa aplikasi standar telepon selular dapat mendukung panjang pesan dengan karakter sepanjang 8 bit (panjanng pesan maksimum 140 karakter) dan karakter yang lainnya seperti 16 bit namun karakter sepanjang 8 bit dan 16 bit ini tidak didukung oleh semua aplikasi standar telepon selular. Pada umumnya karakter sepanjang 8 bit dan 7 bit digunakan untuk menempilkan data seperti gambar dan symbol saja.

Secara umum sebuah telepon selular hanya dapat melakukan pengiriman satu buah paket SMS dalam satu pesan, namun dengan kemajuan teknologi yang ada sekarang, beberapa telepon selular mampu mengirim beberapa paket SMS dalam satu pesan. Yang dilakukan telepon selular agar dapat melakukan beberapa paket dalam satu pengiriman beberapa paket dalam satu kali pengiriman pesan adalah melakukan konkatinasi jadi sebenarnya hal yang dilakukan sama dengan meengirimkan beberapa pesan hanya saja dengan melakukan konkatinasi, beberapa pesan yang disatukan tersebut dapat terlihat menjadi satu buah pesan. Dengan adanya fitur konkatinasi, sebuah SMS seolah-olah dapat mengirim pesan dengan panjang lebih dari 160 karakter dalam satu buah pesan, namun pada fitur konkatinasi ini dibutuhkan sebuah informasi tambahan pada pesan untuk menyambungkan beberapa pesan menjadi satu buah pesan, oleh karena itu panjang satu buah pesan akan menjadi lebih kecil.

2.2 Kriptografi

Kriptografi (*cryptography*) berasal dari bahasa Yunani, terdiri dari dua suku kata yaitu *kripto* dan *graphia*. *Kripto* artinya menyembunyikan, sedangkan *graphia* artinya tulisan. Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi, seperti kerahasiaan data, keabsahan data, integritas data, serta autentikasi data. Tetapi tidak semua aspek keamanan informasi dapat diselesaikan dengan kriptografi. Kriptografi dapat pula diartikan sebagai ilmu atau seni untuk menjaga keamanan pesan. Ketika suatu pesan dikirim dari suatu tempat ke tempat lain, isi pesan tersebut mungkin dapat disadap oleh pihak lain yang tidak berhak untuk mengetahui isi pesan tersebut. Untuk menjaga pesan, maka pesan tersebut dapat diubah menjadi suatu kode yang tidak dapat dimengerti oleh pihak lain.

Enkripsi adalah sebuah proses penyandian yang melakukan perubahan sebuah kode (pesan) dari yang bias dimengerti *(plainteks)* menjadi sebuah kode yang tidak biasa dimengerti *(cipherteks)*. Sedangkan proses kebalikannya untuk mengubah *cipherteks* menjadi *plainteks* disebut *deskripsi*. Proses enkripsi dan deskripsi memrerlukan suatu mekanisme kunci tertentu.

Kriptoanalisis (cryptanalysis) adalah kebalikan dari kriptografi, yaitu suatu ilmu untuk memecahkan mekanisme kriptografi dengan cara mendapatkan kunci dari cipherteks yang digunakan untuk mendapatkan *plainteks*. *Kriptologi (cryptology)* adalah ilmu yang mencakup kriptografi dan kriptoanalisis.

Ada 4 tujuan dari kriptografi yang juga merupakan aspek keamanan informasi seperti :

1. Kerahasian

   Yaitu untuk menjaga isi dari informasi dari siapapun kecuali yang memiliki otoritas atau kunci rahasia untuk membuka informasi yang telah dienkripsi.

2. Integritas Data

   Yaitu untuk menjaga keaslian data dari perubahan data secara tidak sah. Untuk menjaga integritas data, system harus memiliki kemampuan untuk mendeteksi manipulasi data oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, dan pensubsitusian data lain kedalam data yang sebenarnya.

3. Autentikasi

   Ini berhubungan dengan identifikasi/pengenalan, baik secara kesatuan system maupun informasi itu sendri. Dua pihak yang saling berkomunikasi harus saling memperkenalkan diri. Informasi yang dikirimkan melalui kanal harus diautentikasi keaslian, isi datanya, waktu pengiriman, dan lain-lain.

4. Non-repudasi

   Adalah usaha untuk mencegah terjadinya penyangkalan terhadap pengiriman suatu informasi oleh yang mengirimkan, atau harus dapat membuktikan bahwa suatu pesan berasal dari seseorang apabila ia menyangkal hal tersebut.

2.2.1 Sejarah Kriptografi

Kriptografi sudah digunakan sekitar 40 abad yang lalu oleh orang-orang Mesir untuk mengirim pesan ke pasukan yang berbeda di medan perang dan agar pesan tersebut tidak terbaca oleh pihak musuh walaupun pembawa pesn tersebut tertangkap oleh musuh. Sekitar 400 SM, kriptografi digunakan oleh bangsa Spartan dalam bentuk sepotong papirus atau perkamen yang dibungkus dengan batang kayu.

Pada zaman Romawi kuno, ketika Julius Caesar ingin mengirimkan pesan rahasia pada seorang Jendral di medan perang. Pesan tersebut harus dikirimkan melalui seorang prajurit, tetapi karena pesan tersebut mengandung rahasia, Julius Caesar tidak ingin pesan tersebut terbuka di tengah jalan. Di sini Julius Caesar memikirkan bagaimana mengatasinya yaitu dengan mengacak isi pesan tersebut menjadi suatu pesan yang tidak dapat dipahami oleh siapapun kecuali hanya dapat dipahami oleh Jendralnya saja. Tentu sang Jendral telah diberi tahu sebelumnya bagaimana cara membaca pesan yang teracak tersebut, karena telah mengetahui kuncinya.

Pada perang dunia kedua, Jerman menggunakan mesin enigma atau juga disebut dengan mesin rotor yang digunakan Hitler untuk mengirim pesan kepada tentaranya di medan perang. Jerman sangat percaya bahwa pesan yang dienkripsi menggunakan enigma tidak dapat dipecahkan. Tapi anggapan itu keliru, setelah bertahun-tahun sekutu mempelajarinya dan berhasil memecahkan kode-kode tersebut. Setelah Jerman mengetahui bahwa enigma dapat dipecahkan, maka enigma mengalami beberapa kali perubahan. Enigma yang digunakan Jerman dapat mengenkripsi suatu pesan sehingga mempunyai 15×10^{18} kemungkinan untuk dapat mendekripsi pesan.

Perkembangan komputer dan sistem komunikasi pada tahun 60-an berdampak pada permintaan dari pihak-pihak tertentu sebagai sarana untuk melindungi informasi dalam bentuk digital dan untuk menyediakan layanan keamanan. Dimulai dari usaha Feistel dari IBM di awal tahun 70-an dan mencapai puncaknya pada 1977 dengan pengangkatan DES (*Data Encryption Standard*) sebagai standar pemrosesan informasi federal Amerika Serikat untuk mengenkripsi informasi yang tidak belum diklasifikasi. DES merupakan mekanisme kriptografi yang paling dikenal sepanjang sejarah.

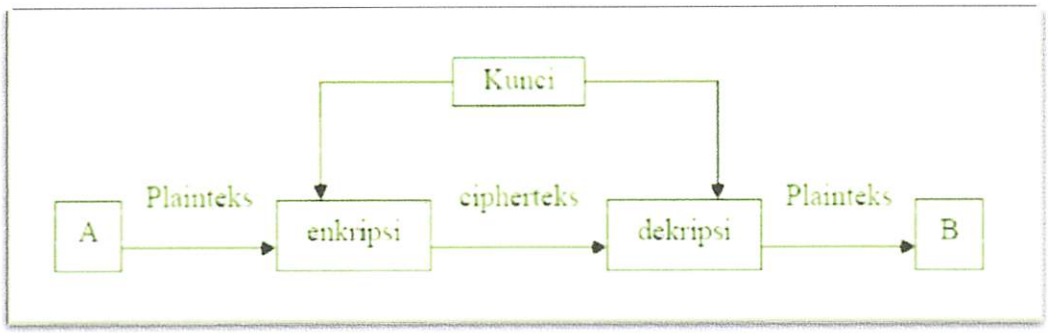
Pengembangan paling mengejutkan dalam sejarah kriptografi terjadi pada 1976 saat Diffie dan Hellman mempublikasikan *"New Directions in Cryptography"*. Tulisan ini memperkenalkan konsep revolusioner kriptografi kunci publik dan juga memberikan metode baru untuk pertukaran kunci, keamanan yang berdasar pada kekuatan masalah logaritma diskret. Meskipun Diffie dan Hellman tidak memiliki realisasi praktis pada ide enkripsi kunci publik saat itu, idenya sangat jelas dan menumbuhkan ketertarikan yang luas pada komunitas kriptografi. Pada 1978 Rivest, Shamir dan Adleman menemukan rancangan enkripsi kunci publik yang sekarang disebut RSA. Rancangan RSA berdasar pada masalah faktorisasi bilangan yang sulit, dan menggiatkan kembali usaha untuk menemukan metode yang lebih efisien untuk pemfaktoran.[3]

2.2.2 Algoritma Kriptografi

Definisi terminologi *algoritma* adalah urutan langkah-langkah logis untuk menyelesaikan masalah yang disusun secara sistematis. Algoritma kriptografi merupakan langkah-langkah logis bagaimana menyembunyikan pesan dari orang-orang yang tidak berhak atas pesan tersebut.

2.2.2.1 Algoritma Simetris

Algoritma ini sering disebut dengan algoritma klasik karena memakai kunci yang sama untuk kegiatan enkripsi dan dekripsi. Algoritma ini sudah ada sejak lebih dari 4000 tahun yang lalu. Bila mengirim pesan dengan menggunakan algoritma ini, si penerima pesan harus diberitahu kunci dari pesan tersebut agar bisa mendekripsikan pesan yang dikirim. Keamanan dari pesan yang menggunakan algoritma ini tergantung pada kunci. Jika kunci tersebut diketahui orang maka orang tersebut dapat melakukan enkripsi dan dekripsi terhadap pesan.



Gambar 2.1. Skema Algoritma Simetris

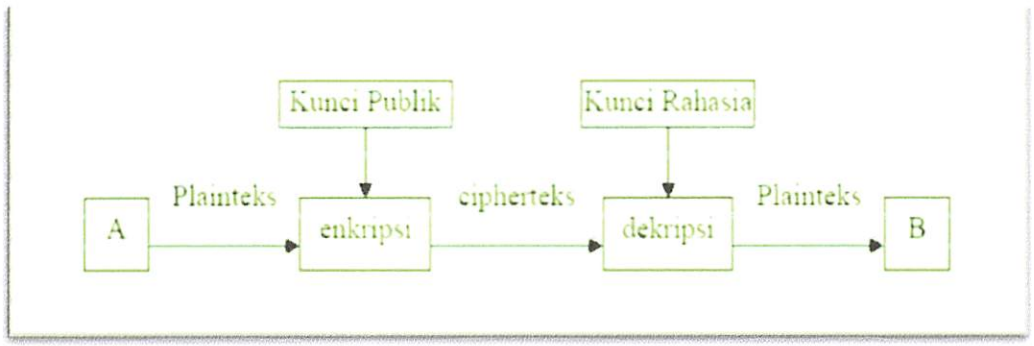
Contoh dari algoritma kriptografi simetris adalah Cipher Permutasi, Cipher Substitusi, Cipher Hill, OTP, RC6, Twofish, Magenta, FEAL, SAFER, LOKI, CAST, Rijndael (AES), Blowfish, GOST, A5, Kasumi, DES dan IDEA.

2.2.2.2 Algoritma Asimetris

Algoritma asimetris sering juga disebut dengan algoritma kunci public, dengan arti kata kunci yang digunakan untuk melakukan enkripsi dan dekripsi berbeda. Pada algoritma asimetris kunci terbagi menjadi 2 bagian, yaitu:

- Kunci umum *(public key)* : yaitu yang boleh semua orang tau (dipublikasikan).
- Kunci rahasia *(private key)* : yaitu kunci yang dirahasiakan (hanya boleh diketahui oleh satu orang).

Kunci-kunci tersebut berhubungan satu sama lain. Dengan kunci public orang dapat mengenkripsi pesan tetapi tidak bisa mendekripsinya. Hanya orang yang memiliki kunci rahasia yang dapat mendeskripsi pesan tersebut. Algoritma asimetri bisa mengirimkan pesan lebih aman daripada algoritma simetri.



Gambar 2.2. Skema Algoritma Asimetris

Contoh dari algoritma asimetris adalah RSA, ElGamal, McEliece, LUC dan DSA (*Digital Signature Algorithm*), Diffie-Hellman (DH), Kriptografi Quantum, dan lain sebagainya.

2.3 RC6 (*Rivest Code* 6)

RC6 merupakan suatu algoritma kriptografi *block cipher* yang dirancang oleh Ronal L.Rivest, Matt J.B. Robshaw, Ray Sidney dan Yuqin Li Yin dari RSA Laboratories. Algoritma ini mulanya dirancang untuk menjadi AES (*Advanced Encryption Standard*). Algoritma RC6 ini berhasil menjadi finalis dan menjadi kandidat kuat untuk menjadi AES walaupun pada akhirnya algoritma ini tidak terpilih menjadi AES melainkan algoritma *Rinjadael.* RC6 mulai dipublikasikan pada tahun 1998. Dasar desain dari algoritma RC6 didasarkan pada pendahulunya. Seperti halnya RC5, parameter dari algortima ini adalah ukuran blok, ukuran kunci eksternal dan jumlah putaran yang bervariasi dengan batasan yang sama.

RC6 dirancang untuk menghilangkan segala ketidakamanan yang ditemukan pada RC5 karena analisis pada RC5 menunjukkan bahwa ternyata jumlah rotasi yang terjadi pada RC5 tidak sepenuhnya bergantung pada data yang terdapat dalam blok. Selain itu, serangan kriptanalisis diferensial juga ternyata dapat menembus keamanan yang ditawarkan RC5. RC6 memiliki kemampuan untuk beroperasi pada mode blok 128 bit. Jika besar blok 128 bit langsung dipaksakan untuk diimplementasikan dengan algoritma RC5, maka akan dibutuhkan register kerja 64 bit. Spesifikasi arsitektur dan bahasa yang menjadi tempat implementasi algoritma yang ditentukan oleh AES belum mendukung pengoperasian 64 bit yang efisien. Oleh karena itu, daripada menggunakan 2 register 64 bit seperti pada RC5, RC6 menggunakan 4 register 32 bit. Cara kerja algoritma RC6 adalah menggunakan 4 buah register dan menggunakan prinsip *Iterated Block Cipher* yang menggunakan iterasi,

Algoritma RC6 adalah versi yang dilengkapi dengan beberapa parameter, sehingga dituliskan sebagai RC6-w/r/b, dimana parameter w merupakan ukuran kata

dalam satuan bit, r adalah bilangan bulat bukan negative yang menunjukan banyaknya iterasi selama proses enkripsi, dan b menunjukan ukuran kunci enkripsi dalam *byte*.

Pada algoritma kriptografi ini, kunci yang digunakan dalam proses deskripsi dan enkripsi merupakan kunci yang sama. Berdasarkan pemrosesan bit, algoritma kunci simetris dibagi menjadi dua bagian, yaitu : algoritma *block chiper* yang melakukan pemrosesan bit per-blok dan algoritma *stream cipher* yang memproses blok secara mengalir per-bit.

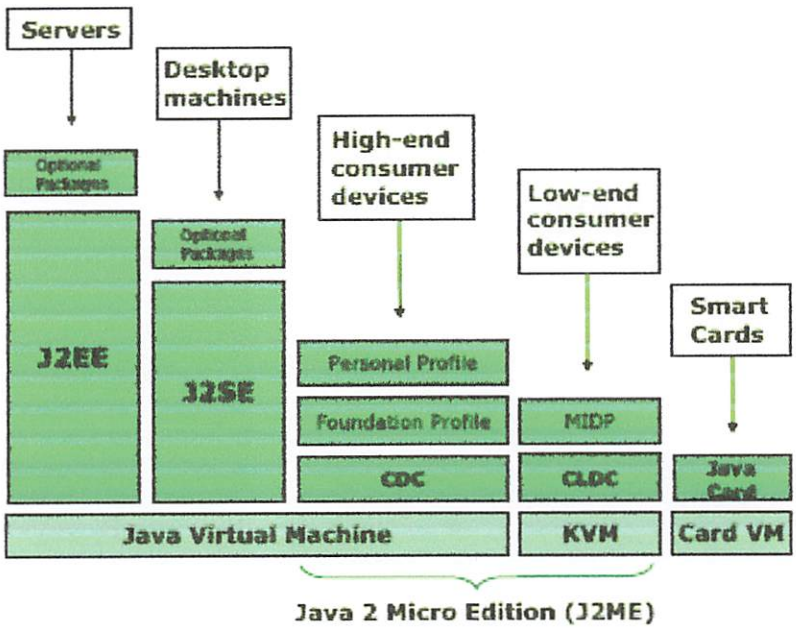
Antara RC6 dengan Rinjdael merupakan algoritma *block cipher* dan sama-sama merupakan kandidat AES, perbandingan algoritma RC6 dengan Rinjdael pada AES sebagai pemenang pada kandidat AES (*Advanced Encryption System*) [5] adalah :

- Kode untuk RC6 lebih sederhana dibandingkan dengan Rinjdael, RC6 juga tidak menghabiskan penggunaan memori yang banyak bila diterapkan kedalam *smart card.*
- RC6 memiliki kelebihan dalam bidang *data depend* dalam enkripsinya dibandingkan Rinjdael.
- Rinjdael memiliki *fleksibilitas* yang tinggi karena dapat diterapkan pada *platform* yang beragam sementara RC6 kurang memiliki kebebasan ini.
- Untuk urusan keamanan, kedua algoritma ini bekerja dalam performa yang baik terlebih karena jumlah putaran semakin mempersulit proses kriptanalisis
- Rinjdael lebih stabil dalam waktu proses enkripsi dan deskripsi, sedangkan RC6 berubah-ubah tergantung sebuah panjang sebuah pesan yang akan di enkripsi dan deskripsi.
- Dibandingkan dengan beberapa algorima yang ada seperti RC4 dan *Blowfish*, maka algoritma RC6 merupakan algoritma enkripsi yang lebih *simple, fast and secure.*

2.4 JAVA

Java menurut definisi dari Sun adalah nama untuk sekumpulan teknologi untuk membuat dan menjalankan perangkat lunak pada komputer *standalone* ataupun pada lingkungan jaringan. Java2 adalah generasi kedua dari Java *platform* (generasi awalnya adalah *Java Development Kit*). Java berdiri di atas sebuah mesin interpreter yang diberi nama *Java Virtual Machine* (JVM). JVM inilah yang akan membaca *bytecode* dalam file `.class` dari suatu program sebagai representasi langsung program yang berisi bahasa mesin. Oleh karena itu, bahasa Java disebut sebagai bahasa pemrograman yang *portable* karena dapat dijalankanpada berbagai sistem operasi, asalkan pada sistem operasi tersebut terdapat JVM.

Platform Java memiliki tiga buah edisi yang berbeda, yaitu J2EE (Java2 *Enterprise Edition*), J2SE (Java2 *Standard Edition*),dan J2ME (Java2 *Micro Edition*). J2EE adalah kelompok dari beberapa API dari Java dan teknologi selain Java. J2EE dibuat untuk membuat aplikasi yang kompleks dan sering dianggap sebagai *middleware* atau teknologi yang berjalan ada *server*. J2SE adalah lingkungan dasar dari Java, sedangkan J2ME akan dibahas pada subbab berikutnya. Ruang lingkup keterhubungan J2EE, J2SE, dan J2ME dapat dilihat pada gambar 2.4

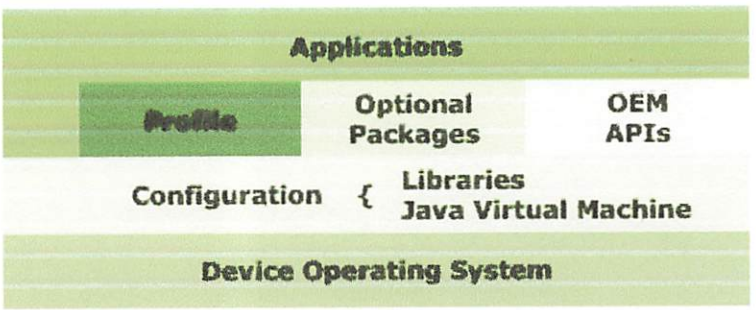


Gambar 2.5 Platform Java

2.4.1 J2ME

Java2 *Micro Edition* atau yang biasa disebut J2ME adalah lingkungan pengembangan yang didesain untuk meletakkan perangkat lunak Java pada barang elektronik beserta perangkat pendukungnya. J2ME membawa Java ke dunia informasi, komunikasi, dan perangkat komputasi selain perangkat komputer *desktop.* J2ME bisa digunakan pada telepon selular(*Handphone*), *pager, Personal Digital Assistance* (PDA) dan sejenisnya.

J2ME adalah bagian dari J2SE, karena itu tidak semua *library* yang ada pada J2SE dapat digunakan pada J2ME. Tetapi J2ME mempunyai beberapa *library* khusus yang tidak dimiliki J2SE. Arsitektur J2ME dapat dilihat pada gambar 2.5.



Gambar 2.6 Arsitektur J2ME

Inti dari J2ME terletak pada *configuration* dan *profile-profile*. Suatu *configuration* menggambarkan lingkungan runtime dasar dari suatu sistem J2ME. Ia menggambarkan *core library, virtual machine*, fitur keamanan dan jaringan. Dalam J2ME telah didefinisikan dua buah konfigurasi yaitu CLDC (*Connected Limited Device Configuration*) untuk perangkat kecil dan CDC (*Connected Decice Configuration*) untuk perangkat yang lebih besar.

2.4.1.1 Configuration

Suatu *configuration* menggambarkan fitur minimal dari lingkungan lengkap Java runtime. Untuk menjamin kemampuan portabilitas dan interoperabilitas optimal diantara berbagai macam perangkat yang dibatasi sumber dayanya(memory, prosesor, koneksi yang dibatasi), *configuration* tidak menggambarkan fitur tambahan. Suatu configuration J2ME menggambarkan suatu komplemen yang minimum dari teknologi Java. Adalah merupakan tugas profile-profile untuk menggambarkan tambahan library untuk suatu kategori perangkat tertentu.

2.4.1.2 *Connected Limited Device Configuration*(CLDC)

CLDC atau *Connected Limited Device Configuration* adalah perangkat dasar dari J2ME, spesifikasi dasar yang berupa *library* dan API yang diimplementasikan pada J2ME, seperti yang digunakan pada telepon selular, PDA dan *pager*. Perangkat tersebut dibatasi dengan keterbatasan memori, sumber daya, dan kemampuan memproses. Spesifikasi CLDC pada J2ME adalah spesifikasi minimal dari *package,* kelas, dan sebagaian *Java Virtual Machine* yang dikurangi agar dapat diimplementasikan dengan keterbatasan sumber daya pada alat-alat tersebut. JVM yang digunakan disebut KVM(*Kilobyte Virtual Machine*).

2.4.1.3 *Connected Device Configuration*(CDC)

CDC atau *Connected Device Configuration* adalah spesifikasi dari konfigurasi J2ME. CDC merupakan komunitas proses pada Java yang memiliki standardisasi. CDC terdiri dari kumpulan *library* dasar untuk dipergunakan pada *profile* industri. *Java Virtual Machine* yang digunakan adalah CDC *Virtual Machine* (CVM). Berikut ini adalah perbandingan antara CLDC dan CDC:

Tabel 2-1. Perbandingan CLDC dan CDC

CLDC	CDC
Mengimplementaikan sebagian dari J2SE	Mengimplementasikan seluruh fitur J2SE
JVM yang digunakan adalah KVM	JVM yang digunakan adalah CVM
Digunakan pada perangkat genggam	Digunakan pada perangkat genggam

(telepon selular, PDA, *pager*) dengan memori terbatas (160-512 KB)	(internet TV, Nokia Communicator, car TV) dengan memori minimal 2MB
Processor : 16 atau 32 bit	*Processor* : 32 bit

2.4.1.4 Profile

Suatu profile menggambarkan set-set tambahan dari API dan fitur untuk pasar tertentu, kategori perangkat atau industri. Sementara *configuration* menggambarkan *library* dasar, profile-profile menggambarkan *library* yang penting untuk membuat aplikasi-aplikasi efektif. *Library* ini memasukkan user *interface*, jaringan dan penyimpanan API.

2.4.1.5 *Mobile Information Device Profile* (MIDP)

MIDP atau *Mobile Device Information Profile* adalah spesifikasi untuk sebuah profil J2ME. MIDP memiliki lapisan di atas CLDC, API tambahan untuk daur hidup aplikasi, antarmuka, jaringan, dan penyimpanan persisten. Pada saat ini terdapat MIDP 1.0 dan MIDP 2.0. Fitur tambahan yang terdapat pada MIDP 2.0 dibanding MIDP 1.0 adalah API untuk multimedia. Pada MIDP 2.0 terdapat dukungan memainkan *tone, tone sequence,* dan *file* WAV walaupun tanpa adanya *Mobile Media API* (MMAPI).

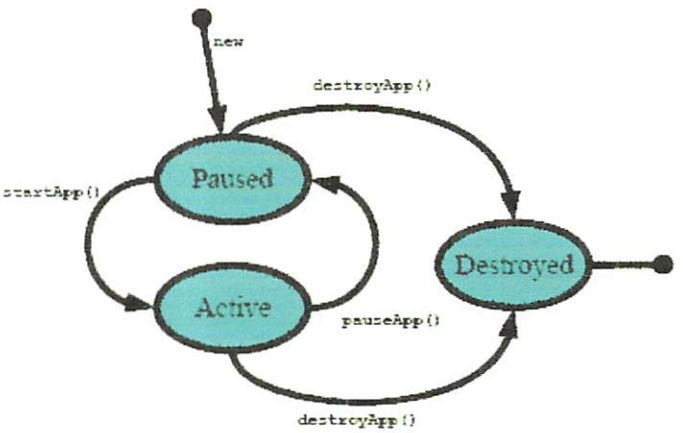
2.4.1.6 MIDlet

Suatu aplikasi MIDP disebut MIDlet. Perangkat *Application Management Software* (AMS) berinteraksi langsung dengan MIDlet dengan method MIDlet create, start, pause, dan destroy. MIDlet adalah bagian dari package `javax.microedition.midlet`. Sebuah

MIDlet harus di-extend dengan class MIDlet. Dan dapat meminta parameter dari AMS seperti dirumuskan dalam application descriptor (JAD).

Suatu MIDlet tidak harus memiliki (dan memang harus tidak mempunyai) sebuah method `public static void main(String[] arg)`. Method tersebut tidak akan dikenal lagi oleh AMS sebagai titik awal sebuah program.

MIDlet terdiri dari beberapa metode yang harus ada, yaitu `constructor()`, `protected void startApp()`, `protected void pause App()`, `protected void destroyApp(boolean unconditional)`. Alur hidup MIDlet dapat dilihat pada gambar 2.7.



Gambar 2.7 Alur Hidup MIDlet

Ketika MIDlet dijalankan maka akan diinisialisasi dengan kondisi *pause* dan dijalankan `pauseApp()`, Kondisi berikutnya adalah fungsi MIDlet dijalankan, yaitu pada `startApp()`. *Metode* yang ada tersebut diimplementasikan sebagai `protected`, hal ini dimaksudkan agar MIDlet lain tidak memanggil *metode* tersebut. Pada saat pemakai keluar dari MIDlet, maka metode `destroyApp()` akan dijalankan kemudian `destroyApp()`akan memanggil *metode* `notifyDestroyed()` sebelum MIDlet benar-benar tidak berjalan lagi. *Metode* `notifyDestroyed()` akan memberitahu *platform* untuk menterminasi MIDlet dan membersihkan semua sumber daya yang mengacu pada MIDlet.

2.4.1.7 JAD (*Java Apllication Descriptor)*

Digunakan untuk mendeskripsikan isi aplikasi untuk keperluan pemetaan. *File* JAD berisi deskripsi *file* JAR (*Java Archive*) dan pemetaan atribut MIDlet, sedangkan *file* JAR berisi kumpulan kelas dan *resource*.[7]

BAB III

DESAIN DAN ANALISA SISTEM

3.1. Spesifikasi Sistem

Aplikasi SMS yang dibangun melakukan komunikasi antara dua *handphone* dimana pada kedua *handphone* tersebut telah di *install* aplikasi SMS yang di desain. Aplikasi itulah yang digunakan untuk proses enkripsi-dekripsi dan pengiriman SMS kepada orang lain. Adapun spesifikasi *hardware* dan *software* pada aplikasi SMS sebagai berikut:

3.1.1. Software

Aplikasi SMS dibangun dengan memanfaatkan IDE NETBEANS 6.5 dan memanfaatkan *plugin Sony Ericsson semc_java_me_cldc_sdk2-5-0-2*

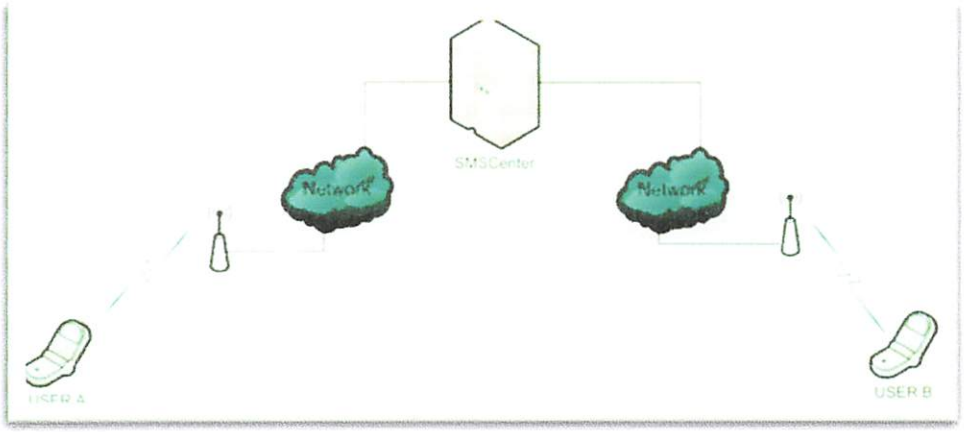
3.1.2. Hardware

Setelah aplikasi yang dibuat telah selesai, kemudian dilakukan implementasi pada *hardware* yang sesungguhnya. Pengujiannya menggunakan *handphone Sony Ericsson W810i* yang sudah di *install* aplikasi. Adapun spesifikasi untuk lingkungan JAVA pada *Sony Ericsson W810i* sebagai berikut:

- CLDC Version : 1.1
- MIDP Version: 2.0

3.2. Desain Keseluruhan Sistem

Perancangan desain dari aplikasi agar dapat dilakukan secara sistematis dan terstruktur maka perlu dibuat blok system yang menjelaskan system yang dirancang. Secara garis besar perancangan desain system ditunjukkan pada gambar berikut:



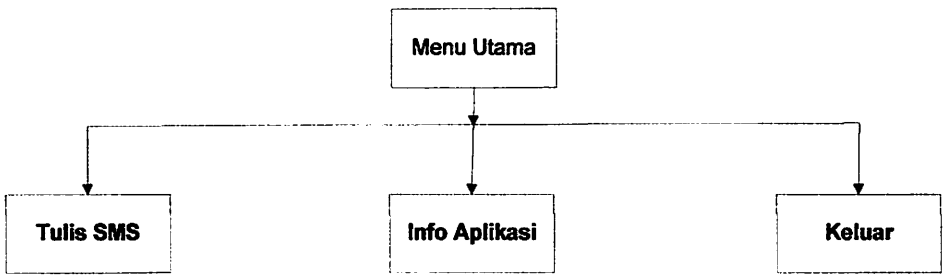
Gambar 3.1 Desain Keseluruhan Sistem

3.3. Desain Aplikasi SMS

SMS (*Short Messaging Service*) adalah sebuah layanan yang dilaksanakan dengan *handphone* untuk mengirim maupun menerima pesan singkat. Pada J2ME diizinkan mengirim dan menerima SMS, namun dengan alasan keamanan sebuah MIDlet hanya dapat memproses pesan SMS yang dikirimkan pada *port* yang terdaftar sebagai *listener.* MIDlet tidak dapat mengakses pesan SMS dari aplikasi lain ataupun yang dikirimkan pada *port* standar (*default*), hal ini sangat berdampak pada penerimaan SMS melalui MIDlet yang pada intinya MIDlet untuk menerima SMS tidak dapat menerima SMS yang masuk ke *inbox handphone*.

Namun MIDlet untuk mengirimkan SMS dapat mengirimkan SMS yang masuk ke *inbox handphone.*

Aplikasi yang dibuat pada intinya mirip dengan menu SMS pada setiap *handphone* pada umumnya, adapun menu dari aplikasi tersebut sebagai berikut:

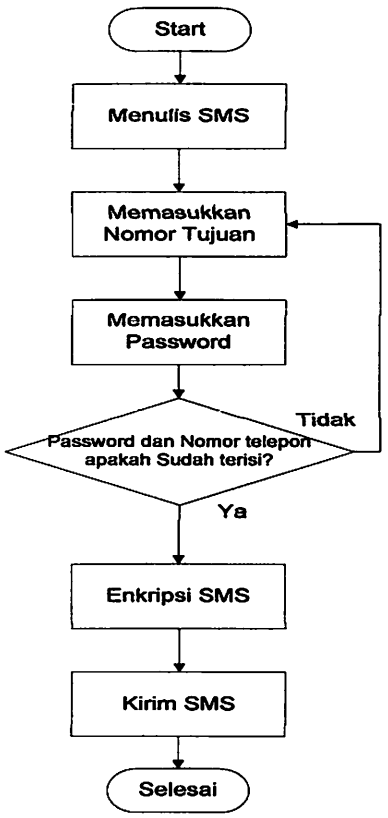


Gambar 3.2 Desain Menu Aplikasi

Untuk mengirim SMS *user* dapat memanfaatkan menu Tulis SMS. Untuk membuat dan mengirim SMS terdapat 3 tahap, yaitu:

- Tulis SMS
- Memasukkan Nomor Tujuan
- Memasukkan *Password*

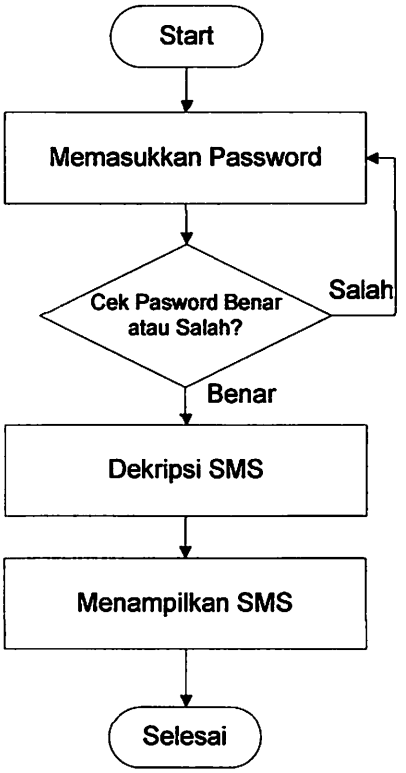
Jika semua tahap telah di jalankan maka user dapat mengirimkan SMS tersebut ke nomor tujuan. Password yang digunakan minimal terdiri dari 1 karakter dan maksimal 16 karakter agar isi pesan tersebut bisa di enkripsi dan panjang *password* maksimal adalah 16. *Flowchart* dari Tulis SMS adalah sebagai berikut:



Gambar 3.3. *Flowchart* Mengirimkan SMS

Penerima SMS tidak harus selalu menjalankan aplikasi ini, karena aplikasi ini memanfaatkan *push registry.* Push Registry berfungsi agar MIDlet bisa meregister koneksi yang masuk dengan Application Management Software (AMS). Jika program tidak berjalan, AMS akan mendengarkan koneksi pada alamat yang telah diregister oleh aplikasi. Hampir semua tipe koneksi didukung, termasuk `ServerSocket` dan `MessageConnection`.

Setelah aplikasi telah berjalan, *user* akan dimintai *password* untuk membuka SMS, jika terdapat kesalahan dalam memasukkan *password* maka isi SMS tersebut tidak akan bisa di baca. Flowchart nya sebagai berikut:

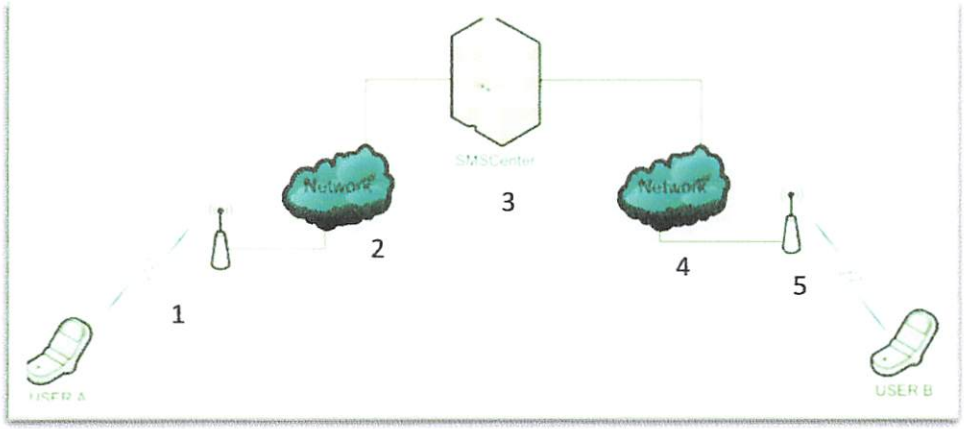


Gambar 3.4. *Flowchart* Menerima SMS

Pada menu setting, user dapat merubah port yang digunakan untuk mengirim pesan, sedangkan menu info aplikasi user digunakan untuk melihat info dari aplikasi ini dan untuk keluar dari aplikasi ini dapat memilih menu keluar.

3.3.1. Analisa Hardware

Analisa Hardware ini membahas aliran koneksi secara hardware dimana aliran SMS melewati beberapa proses sebelum diterima oleh MIDlet tujuan, adapun aliran sebagai berikut:



Gambar 3.5 Analisa Komunikasi SMS

1. *User* A menulis SMS lalu mengirimkannya.
2. Pada *network provider* SMS ditujukan menuju SMSC.
3. Pada SMSC, SMS disimpan menurut setting waktunya. Kemudian SMSC mengirim SMSC tersebut ke nomor *handphone User* B melalui *network provider*.
4. Pada *network provider,* SMS diarahkan menuju ke Nomor *Handphone user* B.
5. SMS diterima oleh *user* B.

3.3.2. Analisa Software

WMA (*Wireless Messaging API*) adalah paket opsional yang terdapat pada J2ME yang mampu mengizinkan kita untuk mengembangkan aplikasi-aplikasi yang mampu melakukan pengiriman dan penerimaan pesan(baik yang berupa teks maupun yang berupa gambar) melalui SMSC(*Short Messaging Service Center*). Kelas-kelas dalam paket ini tersimpan dalam paket `javax.microedition.messaging`.

Pada dasarnya, inti dari paket WMA berada pada *interface* `MessageConnection`, yang merepresentasikan sebuah koneksi jaringan untuk melakukan proses pengiriman maupun penerimaan pesan. Untuk mendapatkan *instance* dari `MessageConnection` dapat dilakukan dengan cara melewatkan URL tertentu ke dalam *method* `Connector.open()`. Berikut ini aturan penulisan URL untuk membuat aplikasi SMS.

- **sms://no_telepon:port.** `MessageConnection` akan mengirimkan pesan ke nomor telepon tujuan untuk tujuan *port* yang ditentukan. Dari sini pesan tidak akan terkirim ke *inbox* SMS *device* yang bersangkutan, melainkan akan dikirimkan ke suatu MIDlet pada *device* penerima yang bertugas mendengarkan *port* tersebut.
- **sms://:port.** `MessageConnection` akan mendengarkan port yang ditentukan. Di sini, MIDlet SMS yang berada di *client* berperan sebagai *server* pada *port* tertentu. Koneksi jenis ini dinamakan *server mode connection.*

Untuk melakukan pengiriman pesan dengan menggunakan J2ME kode yang diperlukan adalah sebagai berikut:

```
String address = "sms://08180808080:5000";
MessageConnection conSend =
          (MessageConnection) Connector.open(address);
TextMessage msg =
   (TextMessage)con.newMessage(MessageConnection.TEXT_MESSAGE);
msg.setAddress(address);
msg.setPayloadText("Ini Aplikasi SMS");
conSend.send(msg);
```

Kode di atas akan mengirimkan SMS dengan teks "Ini Aplikasi SMS" ke nomor tujuan 08180808080 dengan *port* tujuan 5000. Untuk mengirim suatu pesan maka terlebih dahulu membuat obyek `MessageConnection` dengan menuliskan perintah

`Connector.open()`. Setelah itu membuat obyek `TextMessage` dengan memanggil *method* `newMessage()`. Setelah objek TextMessage telah terbentuk, dilakukan pengesetan alamat tujuan dengan *method* `setAddress()` dan juga teks pesan yang akan dikirim menggunakan *method* `setPayloadText()`. Setelah semuanya terpenuhi, lalu memanggil *method* send() untuk mengirimkan obyek `TextMessage` bersangkutan.

Penempatan kode untuk menerima SMS diletakkan dalam objek `Thread` tersendiri yang terpisah. Adapun *method* yang digunakan untuk menerima pesan adalah *method* `receive()` dari objek `MessageConnection`. Dalam pembuatan aplikasi yang dapat menerima SMS perlu di implementasikan *interface* `MessageListener`, yaitu dengan cara mendefinisikan ulang *method* `notifyIncomingMessage()`. *Method* tersebut memiliki satu buah parameter bertipe MessageConnection dan akan dieksekusi pada saat terdapat SMS yang masuk. Berikut ini contoh potongan kode yang diperlukan untuk menerima SMS.

```
String port = "5000";
String address = "...:"+port;
MessageConnection con =
     (MessageConnection) Connector.open( );
Message msg = con.receive();
if(msg instanceof TextMessage){
                TextMessage tMsg = (TextMessage)msg;
     String textSMS = tMsg.getPayloadText();
```

masuk masih bertipe Message sehingga perlu dilakukan *typecasting* terhadap pesan tersebut ke tipe `TextMessage`. Untuk mengambil teks dari pesan yang diterima, digunakan *method* `getPayloadText()` dari objek `TextMessage`.[8]

3.3.3. Analisa Enkripsi - Dekripsi RC 6 Menggunakan *library Bouncycastle*

Bouncycastle merupakan *library* untuk melakukan proses kriptografi menggunakan bahasa pemrograman JAVA atau C++. Di dalam *Bouncycastle* terdapat *class – class* dari algoritma kriptografi yang dapat digunakan untuk melakukan proses kriptografi. Algoritma kriptografi yang sudah ada pada *library Bouncycastle* diantaranya: RC2, RC4, RC5, RC6, *Blowfish,* GOSH, AES, RSA, *Rijndael,*MD5, dsb.

Untuk dapat memanfaatkan *class – class* yang terdapat pada *bouncycastle,* maka pada awal *class* yang kita buat diberi:

```
import org.bouncycastle.crypto.BlockCipher;
import org.bouncycastle.crypto.BufferedBlockCipher;
import org.bouncycastle.crypto.CryptoException;
import org.bouncycastle.crypto.modes.CBCBlockCipher;
import org.bouncycastle.crypto.modes.PaddedBlockCipher;
import org.bouncycastle.crypto.params.KeyParameter;
```

Agar SMS menjadi rahasia maka sebelum dikirim harus diubah menjadi bentuk *ciphertext* terlebih dahulu, adapun bentuk kode nya adalah sebagai berikut:

```
String H = null;
RC6Engine kri1 = new RC6Engine();
BufferedBlockCipher bbc = new PaddedBlockCipher(
new CBCBlockCipher(kri1));
byte[] key = Password.getString().getBytes();
bbc.init(true, new KeyParameter(key));
byte[] chip1=new byte[bbc.getOutputSize(Plain2Chip.getBytes().length)];
int outputLen = bbc.processBytes(Plain2Chip.getBytes(), 0,
                          Plain2Chip.getBytes().length, chip1, 0);
try{
                    bbc.doFinal(chip1, outputLen);
     H=RC6Engine.bytesToHex(chip1);
     System.out.println("Hasil: "+H);
}catch (CryptoException ce){
     System.err.println(ce);
}
```

Dengan kode di atas, maka isi dari SMS akan dirubah menjadi bentuk *ciphertext*, sehingga orang yang tidak mempunyai aplikasi atau yang sudah punya aplikasi tetapi tidak mengetahui *password* untuk membukanya tidak akan bias membaca SMS tersebut.

Pada algoritma kriptografi yang menggunakan sistim *block cipher*, jika kata yang dimasukkan panjangnya tidak dapat memenuhi block, maka akan dilakukan proses *padding*. Proses *padding* adalah mengisi sisa blok yang kosong dengan karakter *null*. Pada kode di atas ditunjukkan pada baris `BufferedBlockCipher bbc = new PaddedBlockCipher (new CBCBlockCipher(kril));`

Untuk melakukan dekripsi, kode yang digunakan adalah sebagai berikut:

```
if(sms instanceof TextMessage) {
        String isiSMSIn = ((TextMessage)sms).getPayloadText();
        String h = null;
        BlockCipher engine2 = new RC6Engine();
        BufferedBlockCipher cipher2 = new PaddedBlockCipher(new
                            CBCBlockCipher(engine2));
        byte[] key2 = Password.getString().getBytes();
        byte[] input2 = RC6Engine.hexToBytes(isiSMSIn);
        cipher2.init(false, new KeyParameter(key2));
        byte[] cipherText2 = new byte[cipher2.getOutputSize(input2.length)];
        int outputLen2 = cipher2.processBytes(input2, 0,
        input2.length, cipherText2, 0);
        try{
            cipher2.doFinal(cipherText2, outputLen2);
              h = RC6Engine.bytesToHex(cipherText2). substring(0,RC6Engine.
                            bytesToHex(cipherText2).indexOf("00"));
            lihatSMSDatang((new String((RC6Engine.hexToBytes(h)))).toString(),
                    alamatPengirim, waktu);
        }catch (CryptoException ce){
            alert.setString("Password Yang Anda Masukkan Salah");
            disp.setCurrent(alert,SMSBaru);
        }
}
```

Karena RC6 adalah jenis algoritma kriptografi yang memakai sistim *block cipher*, maka jika terjadi kesalahan *password* maka aplikasi akan memunculkan eksepsi bahwa terjadi kesalahan. Hal ini berbeda dengan algoritma kriptografi yang

memakai sistim *stream cipher,* aplikasi akan terus melakukan proses dekripsi meskipun password yang dimasukkan salah.

Karena SMS yang diterima masih terdapat karakter *null* yang ditambahkan pada saat prosess *padding,* maka untuk menghilangkannya dengan menambahkan kode sebagai berikut: `h=RC6Engine.bytesToHex(cipherText2).substring(0,RC6Engine. bytesToHex(cipherText2).indexOf("00"));`

3.3.4. Analisa Enkripsi dan Dekripsi RC6

Algoritma yang digunakan untuk proses enkripsi dan dekripsi pada aplikasi ini adalah RC6 *(Rivest Code 6).* Algoritma ini didasarkan pada hasil permutasi yang acak. Mekanisme yang digunakan pada algoritma RC6 ini adalah pembagian blok dari 128 bit menjadi empat buah blok yang masing-masing besarnya 32 bit. Fungsi f(x)= x(2x+1) yang diikuti dengan pergeseran 5 bit ke kiri digunakan untuk memberikan tingkat keamanan data yang tinggi. Pada RC6 tidak ditemui adanya Feistel Network seperti pada DES, namun fungsi f(x)=x(2x+1) dan pergeseran 5 bit ke kiri sudah dapat menghasilkan tingkat pengacakan data yang tidak kalah dengan algoritma yang menggunakan feistel network.

Pada saat melakukan enkripsi-dekripsi RC6 menggunakan beberapa iterasi. Penentuan banyaknya iterasi didasarkan atas azas keseimbangan. Dimana jika jumlah *round* sedikit akan menyebabkan *ciphertext* menjadi mudah untuk dipecahkan. Sedangkan jika jumlah iterasi semakin banyak, akan menyebabkan kecepatan proses enkripsi-dekripsi akan semakin berkurang.

Sebenarnya jumlah *round* sebanyak 20 cukup beralasan. Akan tetapi berdasarkan analisis keamanan yang sangat mendalam, diketahui serangan terbaik terahadap RC6 dapat mencapai 8 *round.* Untuk alasan ini, pencipta RC6 memeberikan "*security margin*" yang cukup besar yaitu sebanyak 12 *round* untuk memberikan perlindungan keamanan yang tinggi terhadap RC6.

Penjadwalan kunci pada RC6 menggunakan sebuah sBox untuk melakukan iterasi awal terhadap S[0]...S[43]. Nilai konstanta P32=B7E15163 (heksadesimal) didefinisikan sebagai inisialisasi awal terhadap S[0] dan inisialisasi awal S[1] hingga S[43] didapatkan dengan menambahkan Q32=9E3779B9 (heksadesimal) terhadap inisialisasi awal subkunci-subkunci sebelumnya.

Pada proses utama penjadwalan kunci, digunakan beberapa operasi seperti pergeseran bit dan operasi penjumlahan yang menyebabkan proses penjadwalan kunci pada RC6 mempunyai kekuatan yang sama dengan RC6 cipher itu sendiri.

Algoritma RC6 dilengkapi dengan beberapa parameter, sehingga dituliskan sebagai RC6-w/r/b. Parameter w merupakan ukuran kata dalam satuan bit, parameter merupakan bilangan bukan negatif yang menunjukan banyaknya iterasi selama proses enkripsi dan parameter b menunjukan ukuran kunci enkripsi dalam byte. Setelah algoritma ini masuk dalam kandidat AES, maka ditetapkan bahwa nilai w = 32, r=20 dan b bervariasi antara 16, 24 dan 32 byte. RC6-w/r/b memecah blok 128 bit menajdi 4 buah blok 32-bit, dan mengikuti aturan enam operasi dasar sebagai berikut :

a + b operasi penjumlahan bilangan integer

a - b operasi pengurangan bilangan integer

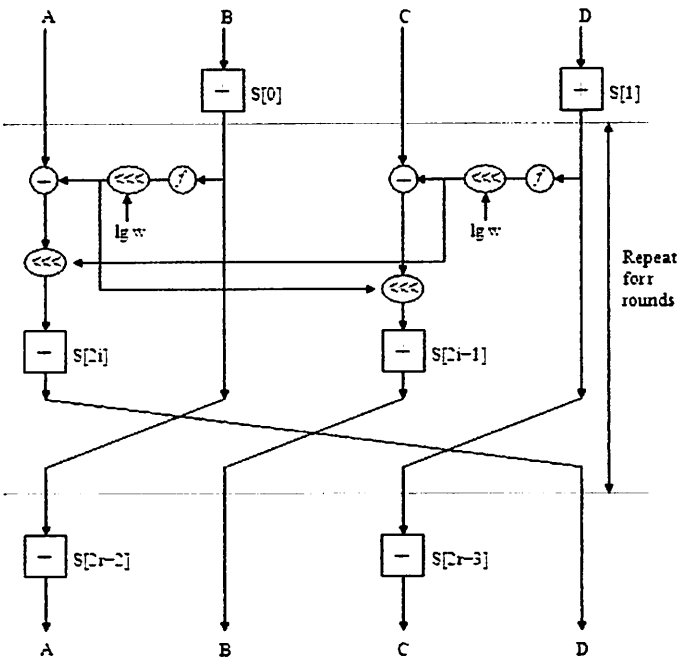
a ⊕ b operasi exclusive-OR (XOR)

a x b operasi perkalian bilangan integer

a<<<b a dirotasikan ke kiri sebanyak variabel kedua (b)

a>>>b a dirotasikan ke kanan sebanyak variabel kedua (b)

Karena RC6 memecah blok 128 bit menjadi 4 buah blok 32 bit, maka algoritma ini bekerja dengan 4 buah register 32-bit A, B, C, D. Byte yang pertama dari plaintext atau ciphertext ditempatkan pada byte A, sedangkan byte yang terakhirnya ditempatkan pada byte D. Dalam prosesnya akan didapatkan (A, B, C, D) = (B, C, D, A) yang diartikan bahwa nilai yang terletak pada sisi kanan berasal dari register disisi kiri. Diagram blok berikut akan lebih menjelaskan proses enkripsi yang terjadi pada algoritma RC6 :



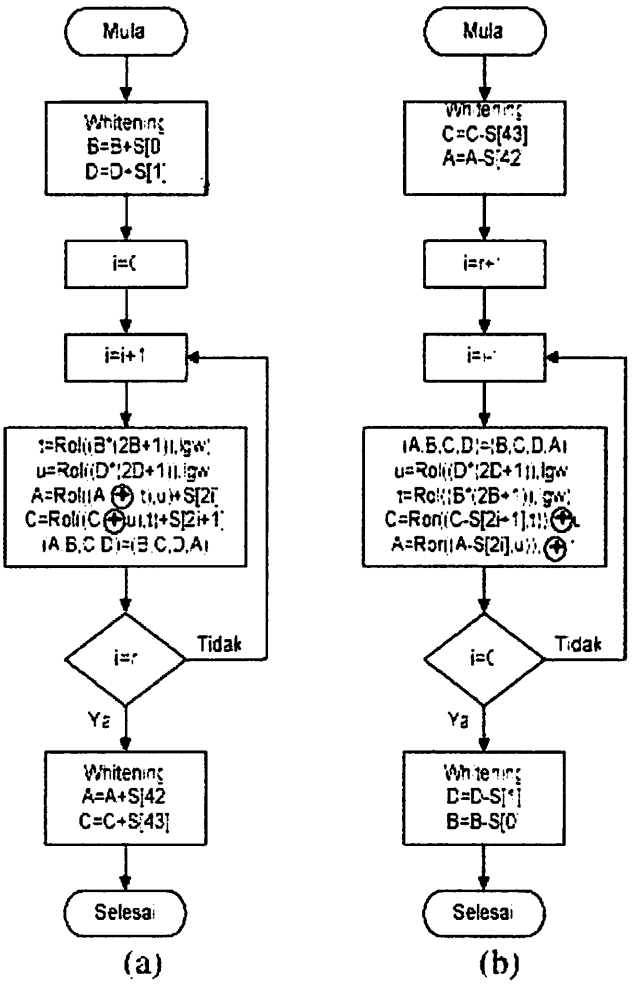
Gambar 3.6 Proses Algoritma RC6

Algoritma RC6 menggunakan 44 buah sub kunci yang dibangkitkan dari kunci dan dinamakan dengan S[0] hingga S[43]. Masing-masing sub kunci panjangnya 32 bit. Proses enkripsi pada algoritma RC6 dimulai dan diakhiri dengan proses *whitening* yang bertujuan untuk menyamarkan iterasi yang pertama dan yang terakhir dari proses enkripsi dan dekripsi. Pada proses whitening awal, nilai B akan dijumlahkan dengan S[0], dan nilai D dijumlahkan dengan S[i]. Pada masing-masing iterasi pada RC6 menggunakan 2 buah sub kunci. Sub kunci pada iterasi yang pertama menggunakan S[2] dan S[3], sedangkan iterasi-iterasi berikutnya menggunakan sub-sub kunci lanjutannya. Setelah iterasi ke-20 selesai, dilakukan proses whitening akhir dimana nilai A dijumlahkan dengan S[42], dan nilai C dijumlahkan dengan S[43].

Setiap iterasi pada algoritma RC6 mengikuti aturan sebagai berikut, nilai B dimasukan ke dalam fungsi f, yang didefinisikan sebagai $f(x) = x(2x+1)$, kemudian diputar kekiri sejauh lg-w atau 5 bit. Hasil yang didapat pada proses ini dimisalkan sebagai u. Nilai u kemudian di XOR dengan C dan hasilnya menjadi nilai C.

Nilai t juga digunakan sebagai acuan bagi C untuk memutar nilainya kekiri. Begitu pula dengan nilai u, juga digunakan sebagai acuan bagi nilai A untuk melakukan proses pemutaran kekiri.

Kemudian sub kunci S[2i] pada iterasi dijumlahkan dengan A, dan sub kunci S[2i+1] dijumlahkan dengan C. keempat bagian dari blok kemudian akan dipertukarkan dengan mengikuti aturan, bahwa nilai A ditempakan pada D, nilai B ditempatkan pada A, nilai C ditempatkan apda B, dan nilai (asli) D ditempatkan pada C. demikian iterasi tersebut akan terus berlangsung hingga 20 kali. Berikut adalah gambar diagram alir (*flowchart*) dari proses enkripsi dan deskripsi secara umum algoritma RC6 :



Gambar 3.7 (a) *Flowchart* untuk proses enkripsi (b) *flowchart* untuk proses deskripsi

3.3.4.1 Pembentukan Kunci Internal

Untuk membangkitkan kunci internal yang akan digunakan pada proses enkripsi, algoritma RC6 melakukan proses pembangunan kunci yang identik dengan algoritma RC5, yang membedakan hanyalah pada algoritma RC6 adalah jumlah *word* yang diambil dari kunci yang dimasukkan oleh pengguna ketika melakukan enkripsi ataupun deskripsi lebih banyak. Tujuan dari proses pembangkitan kunci internal adalah untuk membentuk suatu *array* S yang berukuran 2r+4 dari kunci masukan sepanjang *b bytes* ($0 \leq b \leq 255$), *array* tersebut akan digunakan baik dalam proses enkripsi dan deskripsi.

Proses untuk membangun kunci-kunci internal menggunakan dua buah konstanta yang disebut dengan "*magic constant*" . Dua buah *magic constant* tersebut *Pw* dan *Qw* yang didefiniskan sebagai berikut :

$Pw = \text{Odd}((e - 2) \times 2^w)$

$Qw = \text{Odd}((\text{ø} - 1) \times 2^w)$

Dengan :

e = 2.718281828459..... (basis algoritma natural)

ø = 1.6180339887..... (*golden* ratio)

Sedangkan fungsi *Odd* (x) menghasilkan nilai integer ganjil yang paling deket dengan x. Untuk nilai – nilai *w* yang diperbolehkan (16, 32, dan 64). Berikut adalah daftar *magic constant* dalam heksadesimal :

$P_{16} = b7e1$

$Q_{16} = 9e37$

$P_{32} = b7e15163$

Q_{32} = *9e3779b9*

P_{64} = *b7e151628aed2a6b*

Q_{64} = *9e3779b97f4a7c15*

Dengan menggunakan dua buah *magic constant* tersebut, pembangunan kunci terdiri dari tiga tahap yaitu :

1. Konversi kunci rahasia dari *byte* ke *words*

   Langkah pertama adalah menyalin kunci rahasia K [0...b-1] kedalam sebuah *array* L[0...c-1], dimana c = pembulatan keatas dan u = *w*/8, penyalinan tersebut dilakukan secara *little endian* . Untuk semua posisi *byte* pada L yang kosong diberi nilai nol . Langkah ini dapat dilakukan dengan cara ini :

```
if c=0 then
        c←1
endif
for i←b-1 downto 0 do
        L[i/u] ← (L[i/u]<<<8) + K[i]
endfor
```

2. Inisialisasi *array* S

   Langkah kedua adalah melakukan inisialisasi *array* S agar memiliki pola *pseudorandom* bit tertentu menggunakan modulo 2^w yang ditentukan dengan P_w dan Q_w. Berikut adalah langkah kedua :

```
S[0] ← Pw
for i←1 to 2r+3 do
        S[i]←S[i-1]+ Qw
endfor
```

3. Menyatukan L dan S

Terakhir menyatukan kunci rahasia dari pengguna yang sudah tersimpan dalam L dan S sebanyak 3 kali iterasi . Berikut adalah tahap menyatukan kunci tersebut :

```
i←0
j←0
A←0
B←0
v←3*max(c,2r+4)
for index←1 to v do
        S[i]←(S[i]+A+B) <<< 3
        A←S[i]
        L[j]←(L[j]+A+B) <<< (A+B)
        B←L[j]
        i←(i+1)mod(2r+4)
        j←(j+1)mod c
endfor
```

Pembentukan kunci yang dilakukan , mengubah kunci dari *user* yang panjangnya beragam (0 - 255) menjadi suatu rangkaian kunci dengan sepanjang *word* sebanyak 2r + 3 buah.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1. Implementasi

4.1.1. Implementasi Sistem

Tahap implementasi perangkat lunak merupakan proses pengubahan spesifikasi sistem menjadi sistem yang dapat dijalankan. Tahap ini merupakan kelanjutan dari tahap perancangan, yaitu proses pemrograman perangkat lunak sesuai dengan spesifikasi dan desain sistem.

Aplikasi SMS dibuat menggunakan bahasa pemrograman JAVA dengan spesifikasi J2ME. Program bantu untuk mengimplementasikan desain sistem menggunakan *Netbeans 6.5* dan plugin *mobility pack.*

4.1.2. Proses Algoritma RC6

Untuk melakukan proses algoritma RC6, dapat dipakai *class RC6Enggine* dari *bouncy castle. Class* ini berfungsi melakukan perubahan dari *plaintext* ke *chipertext* dan dari *chipertext* ke *plaintext.* Variabel yang harus dimasukkan kedalam *Class RC6* adalah *password* dan text yang akan dirubah(*plaintext* atau *chipertext*).

4.2. Pengujian

Pengujian dilakukan dengan menggunakan emulator yang sudah terdapat pada *Sony Ericsson semc_java_me_cldc_sdk2-5-0-2.* Pengujian meliputi, mengirim SMS, menerima SMS, dan menu info aplikasi.

Gambar 4.1. Menu Utama

Penjelasan tentang fungsi dari item-item pada menu utama sebagai berikut:

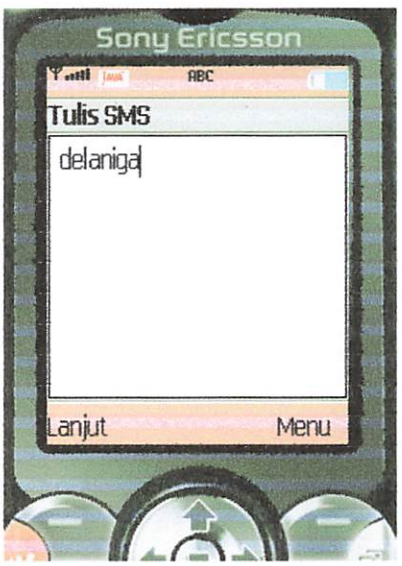
- **Tulis SMS** digunakan untuk membuat pesan baru dan mengirimkannya.
- **Info Aplikasi** untuk mengetahui tentang info dari Aplikasi, seperti Nama MIDlet, Versi, *Vendor,* dan ukuran file.
- **Keluar** untuk keluar dari aplikasi.

4.2.1. Pengujian Mengirim SMS

Pengujian Mengirim SMS dilakukan dengan cara mengirimkan sebuah SMS dan diberi *password.* Pengujian dimulai dari saat pengirim memilih menu "tulis pesan", setelah itu dilanjutkan dengan menulis isi dari SMS.

Setelah selesai menulis isi SMS maka dilanjutkan dengan perintah "Lanjut" sehingga muncul *form* yang berisi *textfield* Nomor Tujuan dan *Password.* Jika isi SMS kosong, maka akan muncul *alert* yang menyatakan bahwa pengirim belum menulis SMS.

Gambar 4.2. Alert Error Jika *Pengirim* Belum Menulis Isi SMS



Gambar 4.3. Menulis Isi SMS

Textfield Nomor Tujuan memiliki batasan (*constraint*) *phonenumber,* maka *textfield* ini hanya dapat diisi dengan nomor telepon saja. Untuk mengisi nomor telepon dapat langsung mengambil dari *phonebook handphone* atau menulis manual. *Textfield Password* memiliki *constraint password,* maka karakter yang dimasukkan oleh pengirim dalam *textfield* ini akan tampil karakter *asterix* untuk alasan keamanan.

Gambar 4.4. Alert Error Memasukkan Nomor Telepon dan Password

Semua *textfield* harus diisi, sehingga jika pengirim tidak mengisi salah satu atau semua maka tidak akan bisa mengirim SMS dan *form* ini akan tampil kembali. Jika *pengirim* telah mengisi Nomor. Tujuan dan *password* maka dapat melanjutkan dengan perintah "Kirim"

Gambar 4.5. Form No. Tujuan dan Password

Setelah itu akan muncul *alert* yang berisi informasi bahwa sedang mengirimkan SMS ke nomor tujuan melalui sebuah port.

Gambar 4.6. Alert Informasi Saat Pengiriman SMS

4.2.2. Pengujian Menerima SMS

Saat penerima menerima notifikasi ada SMS, maka aplikasi akan menanyakan *password* untuk melakukan dekripsi terhadap SMS. Jika *textbox* tidak diisi maka akan muncul *alert* yang berisi peringatan bahwa penerima belum memasukkan *password.*

Gambar 4.7. Alert Error Belum Memasukkan Password

Error jika tidak memasukkan *password* terjadi karena dibutuhkannya password untuk melakukan proses dekripsi. Maka penerima diharuskan

memasukkan *password* yang sama dengan pengirim untuk membaca pesan. *Textbox* Masukkan password mempunyai batasan *password*, maka karakter yang dimasukkan akan tampil sebagai karakter *asterix* dan batas maksimum dari inputan *password* adalah 16 karakter.

Setelah selesai memasukkan *password* penerima dapat menggunakan perintah "Lanjut" untuk melakukan dekripsi dan membaca isi SMS. Jika *password* yang dimasukkan tidak sama dengan yang dimasukkan pengirim saat melakukan enkripsi, maka hasil dekripsi tidak akan sama dengan isi SMS sebenarnya.

Contoh:

Pengirim mengirim SMS dengan isi "delaniga" dengan *password* "123", jika penerima memasukkan *password* selain angka "123" maka hasil dekrisi SMS tidak akan muncul tetapi akan mendapat peringatan bahwa *password* yang dimasukkan salah.

Gambar 4.9. Hasil Dekripsi Jika Password Benar

Untuk membuktikan penerima tidak dapat memasukkan *password* maka hasil yang didapat adalah sebagai berikut :

Gambar 4.10. Alert Jika Password Salah

4.2.3. Pengujian Menu Info Aplikasi

Menu Info Aplikasi berfungsi untuk mengetahui property property yang terdapat pada MIDlet SMS RC6. Pada saat membuka info aplikasi untuk kembali ke menu utama dengan menggunakan perintah "Kembali".

Gambar 4.13. Tampilan Saat Mengeksekusi Menu Info Aplikasi

4.2.4. Pengujian Algoritma RC6 pada Penerimaan Pesan

Pengujian pengiriman dan penerimaan pesan ini bertujuan untuk melakukan pengecekan terhadap pesan yang dikirimkan dan diterima. Pengujian terhadap algoritma RC6 dilakukan dengan mengirim SMS yang dienkripsi lalu dilakukan dekripsi dengan mencoba beberapa *handphone* lain. Tata cara dari pengujian ini adalah melakukan pengiriman terhadap telepon seluler (*handphone*) yang tidak terinstalasi aplikasi yang telah dibangun.

Tabel 4.1 Tabel Pengujian Pengiriman Pesan

No	Pesan yang dikirim	Pesan yang diterima	Jenis Telepon Seluler	Hasil
1.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Samsung Valencia	Dapat menerima pesan berupa *ciphertext*
2.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Blackberry Bold 9000	Dapat menerima pesan berupa *ciphertext*
3.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Nokia E63	Dapat menerima pesan berupa *ciphertext*
4.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Sony Ericsson W810i	Dapat menerima pesan berupa *ciphertext*
5.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Siemens C50	Dapat menerima pesan berupa *ciphertext*
6.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Sony Ericsson W580i	Dapat menerima pesan berupa *ciphertext*
7.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Nokia 5300	Tidak dapat menerima pesan
8.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Nokia 3500c	Tidak dapat menrima pesan
9.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Sony Ericsson K550	Dapat menrima pesan berupa

				ciphertext
10.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Samsung S259 CDMA	Dapat menerima pesan berupa *ciphertext*
11.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Nokia 7610	Dapat menerima pesan berupa *ciphertext*
12.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Nokia 2600c	Tidak dapat menerima pesan
13.	algoritma RC6	9b7ff3f11c87b7d85 e68decbf914d35b	Nokia 5130c	Tidak dapat menerima pesan

Dari tabel 4.1 diatas yang dimaksud dengan pesan tidak dapat diterima adalah aplikasi standar telepon seluler yang menerima pesan tidak mampu menampilkan isi pesan yang bersifat *binary*. Yang dimaksud dengan tidak dapat menerima bukan berarti telepon seluler tersebut tidak dapat menerima pesan terenkripsi melainkan aplikasi standar telepon seluler tersebut tidak mampu menangani penerimaan pesan berbentuk *binary*.

BAB V

PENUTUP

5.1. Kesimpulan

- Dengan menggunakan aplikasi mobile SMS RC6 maka isi SMS dapat dirahasiakan dengan memanfaatkan algoritma RC6 yang lebih *simple, fast and secure.*
- Kelebihan dari algoritma RC6 adalah algoritma enkripsi dengan model *private key*/kunci pribadi yang sama antara proses enkripsi dan proses deskripsinya. Jadi jika *password* yang dimasukan tidak sama maka sebuah pesan tidak dapat dibuka dan dibaca.
- Terdapat kekurangan juga pada Algoritma RC6 adalah penerapan aplikasi yang terbatas pada *platform* tertentu saja.
- Pada pengujian menggunakan *emulator* dan telepon seluler (*handphone*), hasil yang didapat adalah sama.
- Pengujian dilakukan dengan pesan yang pendek dan pesan yang panjang. Panjang maksimal pesan adalah 140 karakter. Dan minimal adalah 1 karakter. Hasilnya yaitu pesan terenkripsi dengan baik.

5.2. Saran

Untuk mengembangkan aplikasi ini dapat dilakukan dengan beberapa cara, diantaranya:

- Untuk mengatasi kekurangan algoritma RC6, dapat digunakan fungsi *hash* seperti MD5 dan SHA.
- Untuk menyimpan SMS masuk, dan SMS keluar, dengan cara menambahkan *Record Management System*(RMS).
- Untuk mempercantik tampilan dapat menggunakan *library* lain, seperti LWUIT(*Light Weight User Interface Toolkit*), *Synclast, Thinlet,* dan *J4ME.*

DAFTAR PUSTAKA

[1] **Ariyus Dony,** *Kriptografi*, Bandung 2008. ITB

[2] **SMS,** http://id.wikipedia.org/wiki/SMS

[3] **JEDI.** *JENI Pengembangan Perangkat Mobile versi 1.1* . JARDIKNAS.2007

[4] **Shalahudin, M.** *Pemrograman J2ME* . Bandung : Penerbit Informatika Bandung.2006.

[5] **Raharjo Budi, Haryono Arif,** *Java,* Bandung 2007. ITB

[6] **Raharjo Budi,** *Mudah Belajar Java,* Bandung 2007. ITB

L A M P I R A N

Institut Teknologi Nasional Malang
Fakultas Teknologi Industri
Jurusan Teknik Elektro S-1

LEMBAR PENGAJUAN JUDUL SKRIPSI
JURUSAN TEKNIK ELEKTRO S-1

Konsentrasi : Teknik Energi Listrik/Teknik Elektronika/Teknik Komputer & Informatika*)

1.	Nama Mahasiswa: DELANIGA FAJARIA RUSLANI			Nim: 06.12.584
2.	Waktu Pengajuan	Tanggal: 01	Bulan: 04	Tahun: 2010
3.	Spesifikasi Judul (berilah tanda silang)**) a. Sistem Tenaga Elektrik b. Energi & Konversi Energi c. Tegangan Tinggi & Pengukuran d. Sistem Kendali Industri e. Elektronika & Komponen f. Elektronika Digital & Komputer g. Elektronika Komunikasi h. lainnya			
4.	Konsultasikan judul sesuai materi bidang ilmu kepada Dosen*) Faisal, ST Ketua Jurusan Ir. F. Yudi Limpraptono,MT NIP. P. 1039500274			
5.	Judul yang diajukan mahasiswa:	PENGEMBANGAN APLIKASI MOBILE UNTUK ENKRIPSI SMS DENGAN ALGORITMA RC 6		
6.	Perubahan judul yang disetujui Dosen sesuai materi bidang ilmu	Desain & Implementasi Aplikasi Mobile untuk Enkripsi SMS dengan Algoritma RC6		
7.	Catatan: Persetujuan Judul skripsi yang dikonsultasikan kepada Dosen materi bidang ilmu	Disetujui 200 Dosen Faisal, ST.		

Perhatian:

1. Formulir pengajuan ini harap dikembalikan kepada jurusan paling lambat satu minggu setelah disetujui kelompok dosen keahlian dengan dilampirkan proposal skripsi beserta persyaratan skripsi sesuai form S-1
2. Keterangan: *) Coret yang tidak perlu
   **) dilingkari a, b, c, ...atau g sesuai bidang keahlian

Form S-2

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura-gura No.2
MALANG

Lampiran : 1 (satu) berkas

Pembimbing Skripsi

Kepada : Yth. Ir. Mimien Mustikawati, MT
Dosen Institut Teknologi Nasional
MALANG

Yang bertanda tangan di bawah ini:

Nama	: DELANIGA FAJARIA RUSLANI
Nim	: 06.12.584
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Komputer & Informatika S-1

Dengan ini mengajukan permohonan, kiranya Bapak / Ibu bersedia menjadi Dosen Pembimbing Utama / ~~Pendamping~~ *), untuk penyusunan Skripsi dengan judul (proposal terlampir) :

DESAIN DAN IMPLEMENTASI APLIKASI MOBILE UNTUK ENKRIPSI SMS DENGAN ALGORITMA RC6

Adapun tugas tersebut sebagai salah satu syarat untuk menempuh Ujian Akhir Sarjana Teknik.

Demikian permohonan ini kami buat.

Malang, April 2010

Hormat Kami,

Mengetahui

Ir. F. Yudi Limpraptono, MT

NIP Y. 1039500274

Delaniga Fajaria Ruslani

0612584

Catatan :
*) Coret yang tidak perlu

Form S-3 a

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura-gura No.2
MALANG

Lampiran : 1 (satu) berkas

Pembimbing Skripsi

Kepada : Yth. Ahmad Faisol, ST
Dosen Institut Teknologi Nasional
MALANG

Yang bertanda tangan di bawah ini:

Nama	: DELANIGA FAJARIA RUSLANI
Nim	: 06.12.584
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Komputer & Informatika S-1

Dengan ini mengajukan permohonan, kiranya Bapak / Ibu bersedia menjadi Dosen Pembimbing ~~Utama~~ / Pendamping *), untuk penyusunan Skripsi dengan judul (proposal terlampir) :

DESAIN DAN IMPLEMENTASI APLIKASI MOBILE UNTUK ENKRIPSI SMS DENGAN ALGORITMA RC6

Adapun tugas tersebut sebagai salah satu syarat untuk menempuh Ujian Akhir Sarjana Teknik.

Demikian permohonan ini kami buat.

Malang, April 2010

Hormat Kami,

Mengetahui

Ir. F. Yudi Limpraptono, MT

NIP Y. 1039500274

Delaniga Fajaria Ruslani

0612584

Catatan :
*) Coret yang tidak perlu

Form S-3 a

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura-gura No.2
MALANG

PERNYATAAN KETERSEDIAAN DALAM PEMBIMBING SKRIPSI

Sesuai permohonan dari Mahasiswa :

Nama	: DELANIGA FAJARIA RUSLANI
Nim	: 0612584
Semester	: VII (Tujuh)
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Komputer & Informatika S-1

Dengan ini menyatakan bersedia / tidak bersedia *) Membimbing Skripsi dari mahasiswa tersebut, dengan judul :

DESAIN DAN IMPLEMENTASI APLIKASI MOBILE UNTUK ENKRIPSI SMS DENGAN ALGORITMA RC6

Demikian surat pernyataan ini kami buat agar dapat dipergunakan seperlunya.

Malang, April 2010

Kami yang membuat pernyataan,

Ir. Mimien Mustikawati, MT

NIP.Y. 1030000352

Catatan :
Setelah disetujui formulir
Diserahkan mahasiswa/ I yang bersangkutan
Kepada Jurusan untuk diproses lebih lanjut
*) Coret yang tidak perlu

Form S-3 b

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura-gura No.2
MALANG

PERNYATAAN KETERSEDIAAN DALAM PEMBIMBING SKRIPSI

Sesuai permohonan dari Mahasiswa :

Nama	: DELANIGA FAJARIA RUSLANI
Nim	: 06.12.584
Semester	: VII (Tujuh)
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Komputer & Informatika S-1

Dengan ini menyatakan bersedia / tidak bersedia *) Membimbing Skripsi dari mahasiswa tersebut, dengan judul :

DESAIN DAN IMPLEMENTASI APLIKASI MOBILE UNTUK ENKRIPSI SMS DENGAN ALGORITMA RC6

Demikian surat pernyataan ini kami buat agar dapat dipergunakan seperlunya.

Malang, April 2010

Kami yang membuat pernyataan,

Ahmad Faisol, ST

NIP.Y.

Catatan :
Setelah disetujui formulir
Diserahkan mahasiswa/ I yang bersangkutan
Kepada Jurusan untuk diproses lebih lanjut
*) Coret yang tidak perlu

Form S-3 b

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura-gura No.2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Delaniga Fajaria Ruslani
Nim : 06.12.584
Masa Bimbingan : 30 April 2010 s/d 30 Oktober 2010
Judul Skripsi : **Desain dan Implementasi Aplikasi Mobile Untuk Enkripsi SMS Dengan Metode Algoritma RC6**

No	Tanggal	Uraian	Paraf Pembimbing
1	16, 19, 22 Juli 2010	Revisi program aplikasi	
2	26 Juli 2010	Demo dan acc program aplikasi	
3	13 Agustus 2010	Revisi makalah Seminar Hasil	
4	14 Agustus 2010	Acc makalah Seminar Hasil	
5	21 Agusutus 2010	Revisi Bab I, II, III, IV, dan V	
6			
7			
8			
9			
10			

Malang, Agustus 2010
Dosen Pembimbing 1

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y. 101.8800.189

Form S-4 b

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan sigura-gura No.2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Delaniga Fajaria Ruslani
Nim : 06.12.584
Masa Bimbingan : 30 April 2010 s/d 30 Oktober 2010
Judul Skripsi : Desain Dan Implementasi Aplikasi Mobile Untuk Enkripsi SMS Dengan Metode Algoritma RC6

No	Tanggal	Uraian	Paraf Pembimbing
1		Demo program	
2		Acc Bab I, II, III, IV, V	
3			
4			
5			
6			
7			
8			
9			
10			

Malang,
Dosen Pembimbing II

(Ahmad Faisol, ST)

Form S-4b

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

Formulir Perbaikan Ujian Skripsi

Dalam pelaksanaan Ujian Skripsi Jenjang Strata 1 Jurusan Teknik Elektro Konsentrasi T. Energi Listrik / T. Elektronika / T.Infokom, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : Delaniga F. R
NIM : 06.12.684
Perbaikan meliputi :

1). Diagram Alir (Flow Chart) dr proses Algoritma Enkripsi RC6.

2). Contoh proses Algoritma Enkripsi RC6
Uis : Bagaimana Enkripsi RC6 secara matematis
Sehingga jika ditulis variabel A → proses Enkripsi → hasil Bagaimana?

Malang, 23 - 8 - 2010

(_____)

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. BendunganSigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
KampusII : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama Mahasiswa : Delaniga Fajaria Ruslani
NIM : 06.12.584
Jurusan : Teknik Elektro S-1
Konsentrasi : Teknik Komputer dan Informatika
Judul Skripsi : *Desain dan Implementasi Aplikasi Mobile Untuk Enkripsi SMS Dengan Metode Algoritma RC6*

Dipertahankan dihadapan tim penguji skripsi jenjang Strata Satu (S-1) pada:
Hari : Senin
Tanggal : 23 Agustus 2010
Dengan Nilai : 77.9 (B+)

PANITIA UJIAN SKRIPSI

KETUA

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y.1018800189

ANGGOTA PENGUJI

Dosen Penguji I

I Komang Somawirata, ST., MT.
NIP.P. 1030100361

Dosen Penguji II

Sotyohadi, ST.
NIP.Y. 1039700309

INSTITUT TEKNOLOGI NASIONAL
FAKULTAS TEKNIK INDUSTRI
JURUSAN TEKNIK ELEKTRO

FORMULIR PERBAIKAN SKRIPSI

Dalam pelaksanaan ujian skripsi jenjang Strata satu (S-1) Jurusan Teknik Elektro konsentrasi Teknik Komputer dan Informatika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

Nama : DELANIGA FAJARIA RUSLANI
NIM : 06.12.584
Jurusan : Teknik Elektro S-1
Konsentrasi : Teknik Komputer dan Informatika
Masa Bimbingan : 30 April 2010 s/d 30 Oktober 2010
Judul Skripsi : Desain dan Implementasi Aplikasi Mobile Untuk Enkripsi SMS Dengan Metode Algortima RC6

Tanggal	Uraian	Paraf
Penguji II 23 Agustus 2010	1. Diagram alir (flowchart) dari proses enkripsi algoritma RC6 2. Contoh proses enkripsi RC6	

Mengetahui,

Dosen Pembimbing I

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y. 1018800189

Dosen Pembimbing II

Ahmad Faisol, ST

Dosen Penguji,

Dosen Penguji II

Sotyohadi, ST
NIP.Y.1039700309

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */



import java.io.IOException;
import java.util.Date;
import javax.microedition.io.Connector;
import javax.microedition.lcdui.*;
import javax.microedition.midlet.*;
import javax.wireless.messaging.*;
import org.bouncycastle.crypto.BlockCipher;
import org.bouncycastle.crypto.BufferedBlockCipher;
import org.bouncycastle.crypto.CryptoException;
import org.bouncycastle.crypto.modes.CBCBlockCipher;
import org.bouncycastle.crypto.modes.PaddedBlockCipher;
import org.bouncycastle.crypto.params.KeyParameter;

/**
 * @author Administrator
 */
public class SMSRC6 extends MIDlet implements CommandListener,
MessageListener, Runnable{
    Display display;
    List menu;
    String[] isiMenu={"Tulis Pesan","Info Aplikasi","Exit"};
    TextBox txtbSMSin, txtbSMSout, txtbSMSout2, txtbPass, txtbPort;
    //TextBox txtbSMSin, txtbSMSout, txtbPass, txtbPort;
    Form form, ttgAplikasi;
    // Form form,ttgAplikasi;
    TextField textnomorTujuan, textPassword, textPort;
     //TextField textNoHP, textPassword, textPort;
    Command cmdBalas, cmdKirim,cmdKirim2, cmdKeluar, cmdBack, cmdClr,
cmdLanjut,cmdLanjut2,cmdPilih;//command yang digunakan
    //ne yg asli tdi rc 6 :Select, Send, Clear, Open, Lanjut, Kembali;
    //NE YG ASLI RC6String password,plain2chip;
    String port, SmsKirim, password, passBuka,plain2chip,
            chip2plain,hasilChip;
    Alert alert;
    Message sms;
    boolean stat;
    Thread thread;
    MessageConnection smskonek;
    StringBuffer sb;
    private Image img1, img2, img3, img4, img5, knc;
    Ticker tickMenu, tickMskPswd;
    String z,zz;


    public SMSRC6() {
        display = Display.getDisplay(this);
        menu = new List("MENU", List.IMPLICIT);
        txtbSMSin = new TextBox("SMS Masuk", null, 250, TextField.ANY);
        txtbSMSout = new TextBox("Tulis SMS", null, 250, TextField.ANY);
        txtbSMSout2 = new TextBox("Tulis SMS", null, 250, TextField.ANY);
```

```
        txtbPass = new TextBox("Masukkan Password", null, 16,
TextField.PASSWORD);
        txtbPort = new TextBox("Port Yang Digunakan", null, 5,
TextField.NUMERIC);
        form = new Form("Send SMS");
        textnomorTujuan = new TextField("No.Tujuan:", null, 14,
TextField.PHONENUMBER);
        textPassword = new TextField("Password:", null, 8,
TextField.PASSWORD);
        cmdBack = new Command("Kembali", Command.BACK, 3);
        cmdBalas = new Command("Balas", Command.SCREEN, 1);
        cmdClr = new Command("Hapus Semua", Command.OK, 2);
        cmdKeluar = new Command("Keluar", Command.EXIT, 1);
        cmdKirim = new Command("Kirim", Command.OK, 1);
        cmdKirim2 = new Command("Kirim", Command.OK, 1);
        cmdLanjut = new Command("Lanjut", Command.OK, 1);
        cmdLanjut2 = new Command("Lanjut", Command.OK, 1);
        cmdPilih = new Command("Pilih", Command.OK, 1);
        tickMenu = new Ticker("Aplikasi SMS dengan Menggunakan Kriptografi
RC6");
        tickMskPswd = new Ticker("Masukkan Password untuk membuka Pesan");
        alert = new Alert(null);
         try {
            img1 = Image.createImage("/img/new.png");
            img2 = Image.createImage("/img/bantu.png");
            img3 = Image.createImage("/img/keluar.png");
           // img4 = Image.createImage("/img/setting.png");
            img5 = Image.createImage("/img/ERROR.png");
            knc = Image.createImage("/img/knc.png");
        } catch (IOException ex) {
            img1 = null;
            img2 = null;
            img3 = null;
            img4 = null;
            img5 = null;
            knc = null;
        }
    }

    public void tampilMenu(){
        menu.deleteAll();
        menu.append("Tulis Pesan", img1);
        menu.append("Info Aplikasi", img2);
       // menu.append("Setting", img4);
        menu.append("Exit", img3);
        menu.setCommandListener(this);
        menu.setTicker(tickMenu);
        display.setCurrent(menu);
    }

    public void newSMS(){
        //txtbSMSout.removeCommand(cmdBalas);
        txtbSMSout.addCommand(cmdLanjut);
        txtbSMSout.addCommand(cmdClr);
        txtbSMSout.addCommand(cmdBack);
        txtbSMSout.setCommandListener(this);
        display.setCurrent(txtbSMSout);
```

```
    public void commandAction(Command c, Displayable d) {
        if(d==menu){
            if(menu.isSelected(0)){
                newSMS();
            }else if(menu.isSelected(1)){
                sb = getAppProperties();
                String infoMidlet = new String( sb.toString());
                Alert help = new Alert("Info",
                        infoMidlet ,
                        null, AlertType.INFO);
                help.setTimeout(Alert.FOREVER);
                help.addCommand(cmdBack);
                display.setCurrent(help);

            }else if(menu.isSelected(2)){
                destroyApp(true);
            }
        }

            if(d==txtbSMSout){
                if(c==cmdLanjut){
                    if(txtbSMSout.size()==0){
                        Alert error = new Alert("Error", "Anda belum menulis
SMS",
                            img5, AlertType.ERROR);
                        display.setCurrent(error,txtbSMSout);
                    }else{
                        form.deleteAll();
                        inputNo();
                }

                }else if(c==cmdClr){
                    txtbSMSout.setString("");
                }else if(c==cmdBack){
                    display.setCurrent(menu);
                }
                else if(c==cmdBalas){
                    balas();
                }
        }
        if(d==txtbSMSout2){
                if(c==cmdLanjut){
                    if(txtbSMSout2.size()==0){
                        Alert error = new Alert("Error", "Anda belum menulis
SMS",
                            img5, AlertType.ERROR);
                        display.setCurrent(error,txtbSMSout2);
                    }else{
                        form.deleteAll();
                        form.removeCommand(cmdLanjut);
                        inputNo2();
                }

                }else if(c==cmdClr){
                    txtbSMSout.setString("");
                }else if(c==cmdBack){
                    display.setCurrent(menu);
```

```
}

public void inputNo(){
    form.deleteAll();
    form.append(textnomorTujuan);
    form.append(textPassword);
    form.addCommand(cmdBack);
    form.addCommand(cmdKirim);
    form.setCommandListener(this);
    display.setCurrent(form);
}

public void inputNo2(){
    form.deleteAll();
    form.append(textnomorTujuan);
    form.append(textPassword);
    form.addCommand(cmdBack);
    form.removeCommand(cmdKirim);
    form.addCommand(cmdKirim2);
    form.setCommandListener(this);
    display.setCurrent(form);
}


public String isiPassword(){
    password = textPassword.getString();
    return password;
}

public void startApp() {
    openConnection();
    tampilMenu();
}

private StringBuffer getAppProperties(){
    StringBuffer temp = new StringBuffer();
    temp.append("Nama Midlet : "+getAppProperty("MIDlet-Name"));
    temp.append("\nVersi : "+getAppProperty("MIDlet-Version"));
    temp.append("\nVendor : "+getAppProperty("MIDlet-Vendor"));
    temp.append("\nUkuran : "+getAppProperty("MIDlet-Jar-Size"));
    return temp;
}

public void pauseApp() {
}

public void destroyApp(boolean unconditional) {
    stat = true;
    thread = null;
    if (smskonek!= null){
        try {
            smskonek.close();
        } catch (IOException ex) {}
    }
    notifyDestroyed();
}
```

```
            }
        }
        if(d==txtbSMSin){
            if(c==cmdBack){
                display.setCurrent(menu);
            }else if(c==cmdBalas){
                //tulisSMS();


            }
        }
        if(d==form){
            if(c==cmdBack){
                form.deleteAll();
                display.setCurrent(txtbSMSout);
            }else if(c==cmdKirim){
                plain2chip = txtbSMSout.getString();
                String a= textnomorTujuan.getString();
                String b= textPassword.getString();
                if(a.length()==0|| b.length()==0){
                    Alert kurang = new Alert("ERROR",
                            "password atau nomor telepon tidak boleh kosong",
                            img5, AlertType.ERROR);
                    display.setCurrent(kurang, form);
                }else{
                    kirim();
                    display.setCurrent(menu);
                    txtbSMSout.setString("");
                }
            }
            if(c==cmdKirim2){
                plain2chip = txtbSMSout2.getString();
                String a= textnomorTujuan.getString();
                String b= textPassword.getString();
                if(a.length()==0|| b.length()==0){
                    Alert kurang = new Alert("ERROR",
                            "password atau nomor telepon tidak boleh kosong",
                            img5, AlertType.ERROR);
                    display.setCurrent(kurang, form);
                }else{
                    kirim();
                    display.setCurrent(menu);
                    txtbSMSout2.setString("");
                }
            }
            if(c==cmdLanjut){
                if(textPassword.getString().equals("")){
                    Alert error = new Alert("Error", "Anda belum memasukkan"
+
                            " password",
                        img5, AlertType.ERROR);
                     display.setCurrent(error,form);
                }else{
                    bukaSMS();
                }
            }
        }
```

```
        if(d==txtbPass){
            if(c==cmdLanjut){
                if(txtbPass.size()==0){
                    Alert error = new Alert("Error", "Anda belum memasukkan"
+
                            " password",
                        img5, AlertType.ERROR);
                    display.setCurrent(error,txtbPass);
                }else{
                    bukaSMS();
                }

            }
        }
        if(d==txtbPort){
            if(c==cmdLanjut){
                new Thread(new Runnable() {
                    public void run(){
                        port = txtbPort.getString();
                        Alert sukses = new Alert("Status Port",
                                "Port Telah diubah menjadi: "+port, null,
                                AlertType.INFO);
                        display.setCurrent(sukses, menu);
                        openConnection();
                    }
                }).start();
            }
        }
    }

    public void kirim(){
        new Thread(new Runnable() {
            public void run(){
                try {
                    isiPassword();
                    String H = null;
                    RC6Engine kri1 = new RC6Engine();
                    BufferedBlockCipher bbc = new PaddedBlockCipher(
                            new CBCBlockCipher(kri1));
                    byte[] key = textPassword.getString().getBytes();

                    bbc.init(true, new KeyParameter(key));
                    byte[] chip1=new
byte[bbc.getOutputSize(plain2chip.getBytes().length)];
                    int outputLen = bbc.processBytes(plain2chip.getBytes(), 0,
                            plain2chip.getBytes().length, chip1, 0);
                    try
                    {
                        bbc.doFinal(chip1, outputLen);
                        H=RC6Engine.bytesToHex(chip1);
                        System.out.println("Hasil: "+H);
                    }
                    catch (CryptoException ce)
                    {
                        System.err.println(ce);
                    }
```

```
                    String isiSMS = H;
                    String noTujuan = textnomorTujuan.getString();
                    String address = "sms://"+noTujuan+ ":"+8000;
                    MessageConnection smsKonekSend = null;
                    Message isiSMSnya = null;
                    smsKonekSend = (MessageConnection)
Connector.open(address);
                    TextMessage txtMessage =
                            (TextMessage)smsKonekSend.newMessage
                            (MessageConnection.TEXT_MESSAGE);
                    txtMessage.setAddress(address);
                    txtMessage.setPayloadText(isiSMS);
                    isiSMSnya = txtMessage;
                    smsKonekSend.send(isiSMSnya);
                    alert.setString("Sedang mengirim ke Nomor: "+noTujuan+
                            " dengan isi: "+isiSMS+" pada port :"+8000);
                    System.out.println("Isi SMS Keluar:" +isiSMS);
                    System.out.println("Isi Password:"
+textPassword.getString().getBytes());
                    alert.setImage(knc);
                    alert.setType(AlertType.INFO);
                    display.setCurrent(alert,menu);
                    textPassword.setString(null);
                    //tampilMenu();
                }catch (Throwable t){
                    t.printStackTrace();
                }
            }
        }).start();
    }

    public void bukaSMS(){
        String alamatPengirim = sms.getAddress();
        Date waktu = sms.getTimestamp();
        if(sms instanceof TextMessage) {
            String isiSMSIn = ((TextMessage)sms).getPayloadText();
            String h = null;
                    BlockCipher engine2 = new RC6Engine();
            BufferedBlockCipher cipher2 = new PaddedBlockCipher(new
                    CBCBlockCipher(engine2));

            System.out.print("Masukkan kunci dekripsi : ");

            byte[] key2 = textPassword.getString().getBytes();
            byte[] input2 = RC6Engine.hexToBytes(isiSMSIn);

            cipher2.init(false, new KeyParameter(key2));
            byte[] cipherText2 = new
byte[cipher2.getOutputSize(input2.length)];
            int outputLen2 = cipher2.processBytes(input2, 0,
                    input2.length, cipherText2, 0);
            try
            {
                cipher2.doFinal(cipherText2, outputLen2);
                 System.out.println("Hex asli:
"+RC6Engine.bytesToHex(cipherText2));
                h = RC6Engine.bytesToHex(cipherText2).
```

```
                        substring(0,RC6Engine.
                        bytesToHex(cipherText2).indexOf("00"));
                System.out.println("Hex:
"+RC6Engine.bytesToHex(cipherText2));
                System.out.println("Hex: "+h);
                System.out.println("Hasil: "+new String(cipherText2));
                System.out.println("Hasil2: "+new String
                        (RC6Engine.hexToBytes(h)));
                lihatSMSDatang((new
String((RC6Engine.hexToBytes(h)))).toString(), alamatPengirim, waktu);
            }
            catch (CryptoException ce)
            {
                System.out.println("Password Yang Anda Masukkan Salah");
                alert.setString("Password Yang Anda Masukkan Salah");
                display.setCurrent(alert,form);
            }


        }
    }

    public void openConnection(){
        String alamatKoneksi = "sms://:"+8000;
        //textPassword.setString("");
        stat = true;
        thread = null;
        try{
            smskonek = (MessageConnection) Connector.open(alamatKoneksi);
            smskonek.setMessageListener(this);
        }catch(IOException ioe){
            ioe.printStackTrace();
        }
        //koneksi = PushRegistry.listConnections(true);
        stat = false;
        thread = new Thread(this);
        thread.start();
    }

    public String formatNo(String no){
        String nNo = null;
        nNo = no.substring(6, no.length());
        return nNo;
    }

    public void masukPassword(){
       String noMasuk = sms.getAddress();
       StringItem si = new StringItem(null, "SMS masuk dari
"+formatNo(noMasuk));
        form.deleteAll();
        form.setTitle("SMS Baru");
        form.append(si);
        form.removeCommand(cmdKirim2);
        form.removeCommand(cmdKirim);
        form.append(textPassword);
        form.addCommand(cmdBack);
        form.addCommand (cmdLanjut);
```

```
    form.setCommandListener(this);
    display.setCurrent(form);

}

public void masukPasswordBalas(){
    form.deleteAll();
    form.setTitle("SMS Baru");
    form.append(textnomorTujuan);
    form.append(textPassword);
    form.addCommand(cmdBack);
    form.addCommand (cmdKirim2);
    form.setCommandListener(this);
    display.setCurrent(form);

}

public void lihatSMSDatang(String msg, String alamat,Date w){
    txtbSMSout.setString("");
    txtbSMSout.setTitle(formatNo(alamat)+","+w);
    txtbSMSout.setString(msg);
    txtbSMSout.addCommand(cmdBalas);
    //txtbSMSout.addCommand(cmdLanjut);
    txtbSMSout.setCommandListener(this);
    display.setCurrent(txtbSMSout);
}

public void balas(){
    txtbSMSout2.removeCommand(cmdBalas);
    txtbSMSout2.setString("");
    txtbSMSout2.setTitle("Tulis Balasan Baru");
    txtbSMSout2.addCommand(cmdLanjut);
    txtbSMSout2.setCommandListener(this);
    display.setCurrent(txtbSMSout2);
}

public void closeConnection(){
    try{
        smskonek.close();
    }catch (IOException e){}
}

public void run(){
    try {
        sms = smskonek.receive();
        masukPassword();
        closeConnection();
        openConnection();
    } catch (IOException ex) {}
}

public void startNewThread(){
    stat = false;
    thread = new Thread(this);
    thread.start();
}
```

```
    public void notifyIncomingMessage(MessageConnection conn) {
        if (thread == null){
            startNewThread();
        }
    }

}
```

```
import org.bouncycastle.crypto.BlockCipher;
import org.bouncycastle.crypto.CipherParameters;
import org.bouncycastle.crypto.DataLengthException;
import org.bouncycastle.crypto.params.KeyParameter;

/**
 * An RC6 engine.
 */
public class RC6Engine
    implements BlockCipher
{
    private static final int wordSize = 32;
    private static final int bytesPerWord = wordSize / 8;

    /*
     * the number of rounds to perform
     */
    private static final int _noRounds = 20;

    /*
     * the expanded key array of size 2*(rounds + 1)
     */
    private int _S[];
    private static final char[] kDigits =
{'0','1','2','3','4','5','6','7','8',
    '9','a','b','c','d','e','f'};
    /*
     * our "magic constants" for wordSize 32
     *
     * Pw = Odd((e-2) * 2^wordsize)
     * Qw = Odd((o-2) * 2^wordsize)
     *
     * where e is the base of natural logarithms (2.718281828...)
     * and o is the golden ratio (1.61803398...)
     */
    private static final int    P32 = 0xb7e15163;
    private static final int    Q32 = 0x9e3779b9;

    private static final int    LGW = 5;        // log2(32)

    private boolean forEncryption;

    /**
     * Create an instance of the RC6 encryption algorithm
     * and set some defaults
```

```
     */
    public RC6Engine()
    {
        _S              = null;
    }

    public String getAlgorithmName()
    {
        return "RC6";
    }

    public int getBlockSize()
    {
        return 4 * bytesPerWord;
    }

    /**
     * initialise a RC5-32 cipher.
     *
     * @param forEncryption whether or not we are for encryption.
     * @param params the parameters required to set up the cipher.
     * @exception IllegalArgumentException if the params argument is
     * inappropriate.
     */
    public void init(
        boolean             forEncryption,
        CipherParameters    params)
    {
        if (!(params instanceof KeyParameter))
        {
            throw new IllegalArgumentException("invalid parameter passed to
RC6 init - " + params.getClass().getName());
        }

        KeyParameter        p = (KeyParameter)params;
        this.forEncryption = forEncryption;
        setKey(p.getKey());
    }

    public int processBlock(
        byte[]  in,
        int     inOff,
        byte[]  out,
        int     outOff)
    {
        int blockSize = getBlockSize();
        if (_S == null)
        {
            throw new IllegalStateException("RC6 engine not initialised");
        }
        if ((inOff + blockSize) > in.length)
        {
            throw new DataLengthException("input buffer too short");
        }
        if ((outOff + blockSize) > out.length)
        {
            throw new DataLengthException("output buffer too short");
```

```
        }

        return (forEncryption)
            ?   encryptBlock(in, inOff, out, outOff)
            :   decryptBlock(in, inOff, out, outOff);
    }

    public void reset()
    {
    }

    /**
     * Re-key the cipher.
     * <p>
     * @param  inKey  the key to be used
     */
    private void setKey(
        byte[]      key)
    {

        //
        // KEY EXPANSION:
        //
        // There are 3 phases to the key expansion.
        //
        // Phase 1:
        //   Copy the secret key K[0...b-1] into an array L[0..c-1] of
        //   c = ceil(b/u), where u = wordSize/8 in little-endian order.
        //   In other words, we fill up L using u consecutive key bytes
        //   of K. Any unfilled byte positions in L are zeroed. In the
        //   case that b = c = 0, set c = 1 and L[0] = 0.
        //
        // compute number of dwords
        int c = (key.length + (bytesPerWord - 1)) / bytesPerWord;
        if (c == 0)
        {
            c = 1;
        }
        int[]   L = new int[(key.length + bytesPerWord - 1) / bytesPerWord];

        // load all key bytes into array of key dwords
        for (int i = key.length - 1; i >= 0; i--)
        {
            L[i / bytesPerWord] = (L[i / bytesPerWord] << 8) + (key[i] &
0xff);
        }

        //
        // Phase 2:
        //   Key schedule is placed in a array of 2+2*ROUNDS+2 = 44 dwords.
        //   Initialize S to a particular fixed pseudo-random bit pattern
        //   using an arithmetic progression modulo 2^wordsize determined
        //   by the magic numbers, Pw & Qw.
        //
        _S            = new int[2+2*_noRounds+2];

        _S[0] = P32;
```

```
    for (int i=1; i < _S.length; i++)
    {
        _S[i] = (_S[i-1] + Q32);
    }

    //
    // Phase 3:
    //   Mix in the user's secret key in 3 passes over the arrays S & L.
    //   The max of the arrays sizes is used as the loop control
    //
    int iter;

    if (L.length > _S.length)
    {
        iter = 3 * L.length;
    }
    else
    {
        iter = 3 * _S.length;
    }

    int A = 0;
    int B = 0;
    int i = 0, j = 0;

    for (int k = 0; k < iter; k++)
    {
        A = _S[i] = rotateLeft(_S[i] + A + B, 3);
        B =  L[j] = rotateLeft(L[j] + A + B, A+B);
        i = (i+1) % _S.length;
        j = (j+1) %  L.length;
    }
}

private int encryptBlock(
    byte[]  in,
    int     inOff,
    byte[]  out,
    int     outOff)
{
    // load A,B,C and D registers from in.
    int A = bytesToWord(in, inOff);
    int B = bytesToWord(in, inOff + bytesPerWord);
    int C = bytesToWord(in, inOff + bytesPerWord*2);
    int D = bytesToWord(in, inOff + bytesPerWord*3);

    // Do pseudo-round #0: pre-whitening of B and D
    B += _S[0];
    D += _S[1];

    // perform round #1,#2 ... #ROUNDS of encryption
    for (int i = 1; i <= _noRounds; i++)
    {
        int t = 0,u = 0;

        t = B*(2*B+1);
        t = rotateLeft(t,5);
```

```
            u = D*(2*D+1);
            u = rotateLeft(u,5);

            A ^= t;
            A = rotateLeft(A,u);
            A += _S[2*i];

            C ^= u;
            C = rotateLeft(C,t);
            C += _S[2*i+1];

            int temp = A;
            A = B;
            B = C;
            C = D;
            D = temp;
        }
        // do pseudo-round #(ROUNDS+1) : post-whitening of A and C
        A += _S[2*_noRounds+2];
        C += _S[2*_noRounds+3];

        // store A, B, C and D registers to out
        wordToBytes(A, out, outOff);
        wordToBytes(B, out, outOff + bytesPerWord);
        wordToBytes(C, out, outOff + bytesPerWord*2);
        wordToBytes(D, out, outOff + bytesPerWord*3);

        return 4 * bytesPerWord;
    }

    private int decryptBlock(
        byte[]  in,
        int     inOff,
        byte[]  out,
        int     outOff)
    {
        // load A,B,C and D registers from out.
        int A = bytesToWord(in, inOff);
        int B = bytesToWord(in, inOff + bytesPerWord);
        int C = bytesToWord(in, inOff + bytesPerWord*2);
        int D = bytesToWord(in, inOff + bytesPerWord*3);

        // Undo pseudo-round #(ROUNDS+1) : post whitening of A and C
        C -= _S[2*_noRounds+3];
        A -= _S[2*_noRounds+2];

        // Undo round #ROUNDS, .., #2,#1 of encryption
        for (int i = _noRounds; i >= 1; i--)
        {
            int t=0,u = 0;

            int temp = D;
            D = C;
            C = B;
            B = A;
            A = temp;
```

```
        t = B*(2*B+1);
        t = rotateLeft(t, LGW);

        u = D*(2*D+1);
        u = rotateLeft(u, LGW);

        C -= _S[2*i+1];
        C = rotateRight(C,t);
        C ^= u;

        A -= _S[2*i];
        A = rotateRight(A,u);
        A ^= t;

    }
    // Undo pseudo-round #0: pre-whitening of B and D
    D -= _S[1];
    B -= _S[0];

    wordToBytes(A, out, outOff);
    wordToBytes(B, out, outOff + bytesPerWord);
    wordToBytes(C, out, outOff + bytesPerWord*2);
    wordToBytes(D, out, outOff + bytesPerWord*3);

    return 4 * bytesPerWord;
}


//////////////////////////////////////////////////////////////
//
// PRIVATE Helper Methods
//
//////////////////////////////////////////////////////////////

/**
 * Perform a left "spin" of the word. The rotation of the given
 * word <em>x</em> is rotated left by <em>y</em> bits.
 * Only the <em>lg(wordSize)</em> low-order bits of <em>y</em>
 * are used to determine the rotation amount. Here it is
 * assumed that the wordsize used is 32.
 * <p>
 * @param  x  word to rotate
 * @param  y    number of bits to rotate % wordSize
 */
private int rotateLeft(int x, int y)
{
    return (x << y) | (x >>> -y);
}

/**
 * Perform a right "spin" of the word. The rotation of the given
 * word <em>x</em> is rotated left by <em>y</em> bits.
 * Only the <em>lg(wordSize)</em> low-order bits of <em>y</em>
 * are used to determine the rotation amount. Here it is
 * assumed that the wordsize used is a power of 2.
 * <p>
```

```
     * @param  x  word to rotate
     * @param  y    number of bits to rotate % wordSize
     */
    private int rotateRight(int x, int y)
    {
        return (x >>> y) | (x << -y);
    }

    public int bytesToWord(
        byte[]  src,
        int     srcOff)
    {
        int    word = 0;

        for (int i = bytesPerWord - 1; i >= 0; i--)
        {
            word = (word << 8) + (src[i + srcOff] & 0xff);
        }

        return word;
    }

    public void wordToBytes(
        int    word,
        byte[]  dst,
        int     dstOff)
    {
        for (int i = 0; i < bytesPerWord; i++)
        {
            dst[i + dstOff] = (byte)word;
            word >>>= 8;
        }
    }

    public static String bytesToHex(byte[] raw) {
        int length = raw.length;
        char[] hex = new char[length * 2];
        for (int i = 0; i < length; i++) {
            int value = (raw[i] + 256) % 256;
            int highIndex = value >> 4;
            int lowIndex = value & 0x0f;
            hex[i * 2 + 0] = kDigits[highIndex];
            hex[i * 2 + 1] = kDigits[lowIndex];
        }
        return (new String(hex)).toString();
    }

    public static byte[] hexToBytes(char[] hex) {
        int length = hex.length / 2;
        byte[] raw = new byte[length];
        for (int i = 0; i < length; i++) {
            int high = Character.digit(hex[i * 2], 16);
            int low = Character.digit(hex[i * 2 + 1], 16);
            int value = (high << 4) | low;
            if (value > 127) value -= 256;
                raw[i] = (byte)value;
        }
```

```
        return raw;
    }

    public static byte[] hexToBytes(String hex){
        return hexToBytes(hex.toCharArray());
    }


}
```