BAB II
TINJAUAN PUSTAKA

II.1 Sistem Keamanan Data

Keamanan data menjadi sangat penting, karena dengan semakin terbukanya informasi setiap orang selalu ingin mendapatkan informasi yang dia inginkan baik dengan cara legal maupun cara tidak legal. Banyak cara yang digunakan oleh orang untuk berusaha mendapatkan informasi tersebut, baik secara pasif maupun aktif. [6]

Keamanan merupakan komponen yang vital dalam komunikasi data elektronis. Masih banyak yang belum menyadari bahwa keamanan (*Securyti*) merupakan sebuah komponen penting yang tidak murah. Teknology kriptografi sangat berperan juga dalam proses komunikasi yang digunakan untuk melakukan enkripsi (pengacakan) data yang ditransaksikan selama perjalanan dari sumber ke tujuan dan juga melakukan dekripsi (menyusun kembali) data yang telah teracak tersebut. Berbagai sistem yang telah dikembangkan adalah seperti sistem *private key* dan *public key*. Penguasaan algoritma-algoritma populer digunakan untuk mengamankan data jaga sangat penting. Contoh-contoh algoritma ini antara lain : DES, IDEA, RC5, RSA dan ECC (*Elliptic Curve Cryptography*). [7]

Dari sisi tindakan pihak yang bertanggung jawab, keamanan jaringan komputer terbagi dua level, yaitu :

1. Keamanan fisik peralatan mulai dari server, terminal / client router sampai dengan *cabling*.
2. Keamanan sistem data sekiranya ada penyeludup yang berhasil mendapatkan akses ke saluran fisik jaringan komputer.

II.2 SMS (*Short Messaging Service*)

SMS merupakan salah satu media yang paling banyak digunakan sekarang ini dikarenakan murah dan prosesnya cepat, langsung kepada tujuan. SMS merupakan salah satu fitur di GSM yang dikembangkan dan distandarisasi oleh ETSI. Meskipun telah banyak fitur-fitur dari GSM seperti EMS, MMS dan GPRS keberadaan jasa dan industri yang menggunakan SMS khususnya semakin lama semakin banyak dijumpai. Hal itu juga didukung oleh faktor *hardware* yang semakin hari semakin terjangkau.

SMS merupakan sebuah sistem pengiriman data dalam paket dengan bandwitch kecil. Dengan karakteristik ini, pengiriman suatu data yang dapat dilakukan dengan efisiensi yang sangat tinggi. Pada awalnya SMS diciptakan untuk menggantikan layanan *paging* dengan menyediakan layanan serupa yang bersifat *two-way messaging* ditambah dengan *notification service*, khususnya untuk *voice mail*. Pada perkembangan selanjutnya, muncul jenis-jenis layanan lain seperti *mail, fax* dan *pagging intergration, interactive banking, information service,*dan integrasi dengan aplikasi berbasis internet. Selain itu juga berkembang layanan wireless seperti *SIM download for active action, debet* dan *profile editing, Wireless Point of Sale* (WPS). Serta layanan aplikasi lapangan seperti *remote reasing, remote sensing* dan *location Base Service* (LBS). Integerasi dengan aplikasi berbasis internet mendorong timbulnya layanan seperti *web-based messaging, gamming* dan *chatting*.

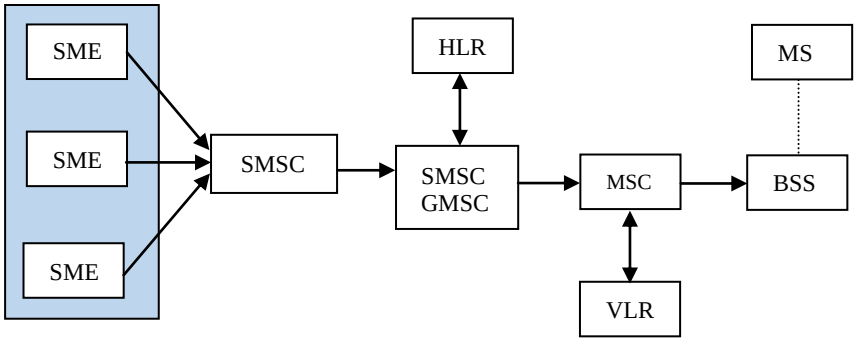
Layanan SMS juga memungkinkan pengiriman pesan dalam bentuk *alphanumeric,* layanan SMS ini banyak diaplikasikan pada sistem komunikasi tanpa kabel (*wireless*). Teknologi *wireless* dipelopori dari kawasan eropa yang

diawali pada kebutuhan bersama terhadap satu sistem jaringan baru yang dapat menjadi standart jaringan yang berlaku dan dapat diterapkan di seluruh kawasan eropa. Dalam sistem baru juga harus terdapat kemampuan yang dapat mengantisipasi mobilitas pengguna untuk menampung penambahan jumlah pelanggan baru.

Pada tahun 1992, dilakukan pengiriman pesan menggunakan SMS dari sebuah *Personal Computer* (PC) ke telepon *mobile* dalam jaringan GSM milik Vodafone Inggris, kemudian merambah ke benua amerika yang di pelopori oleh beberapa operator komunikasi *mobile* berbasis digital seperti *BellSouth Mobility*, *PrimeCo* dan operator lainya. Teknologi yang digunakan dari pengiriman SMS yaitu *Store and forwad service*. [4]

II.2.1 Mekanisme Kerja SMS

SMS mampu mengirim atau menerima data antar jaringan operator seluler secara terus menerus. SMS bukan tidak mengenal batasan wilayah, artinya SMS dapat dikirim atau diterima diseluruh dunia.



Gambar II.1 Elemen-elemen pada Jaringan Operator Seluler
(Sumber : Library.binus.ac.id : 2013)

Short Message Servis Center (SMSC) bertugas untuk menerima dan meneruskan pesan dari telepon selular. SMSC dibangun oleh beberapa *Short*

Message Entyti (SME) yang dapat diletakan dalam dalam sebuah jaringan atau telepon selular. *Mobile Switching Center* (MSC) bertugas mengendalikan koneksi antara telepon seluler dengan jaringan operator seluler. *Gateway Mobile Switching Center* (GMSC) adalah sebuah gerbang MSC yang juga dapat menerima pesan berupa sebuah sistem kontak yang berhubungan dengan jaringan lain.

Dalam menerima pesan dari SMSC, GMSC menggunakan jaringan SS7 (*Signaling System 7*) dalam sistem *Home Location Register* (HLR). HLR adalah *database* utama dalam sebuah jaringan operator seluler. Sistem ini memegang kendali atas informasi nomor-nomor telepon seluler dan juga tentang alur informasi dari setiap nomor telepon seluler, misalnya informasi atas wilayah jangkauan.

Vistor Location Register (VLR) berkorespondensi terhadap setiap MSC. VLR berisi informasi tentang identitas telepon seluler. Dengan bantuan VLR, MSC dapat meneruskan informasi pesan pendek kepada *Base Station System* (BSS), dimana kemudian BSS akan meneruskannya ke telepon seluler penerima.

Sebuah SMS dapat terdiri atas 160 karakter (skema 7 bit), 140 karakter (skema 8 bit) atau 70 karakter (skema 16 bit). Saat ini pengiriman beberapa SMS sekaligus dapat dilakukan. Penggabungan SMS dan kompresi SMS (terdiri lebih dari 160 karakter) telah dikembangkan. Tetapi fitur-fitur ini belum diimplementasikan oleh semua jaingan operator seluler di dunia.

II.2.2 Layanan Aplikasi SMS

Layanan aplikasi SMS merupakan sebuah layanan yang bersifat *none real time* dimana sebuah *short message* dapat di *submit* ke satu tujuan, tidak peduli apakah tujuan tersebut aktif atau tidak. Bila dideteksi bahwa tujuan tidak aktif,

maka sistem akan menunda pengiriman ke tujuan hingga tujuan aktif kembali. Pada dasarnya sistem SMS akan menjamin *delivery* dari suatu *short message* hingga sampai tujuan. [5]

Kegagalan pengiriman yang bersifat sementara seperti tujuan tidak aktif akan selalu teridentifikasi sehingga pengiriman ulang *short message* akan selalu dilakukan kecuali apabila diberlakukan aturan bahwa *short message* yang telah melampaui batas atau waktu tertentu harus dihapus dan dinyatakan gagal kirim. Berdasarkan mekanisme distribusi pesan SMS aplikasi SMS, terdapat 4 macam mekanisme pengaturan pesan yaitu :

1. *Pull,* yaitu pesan yang dikirimkan ke pengguna berdasarkan permintaan pengguna.
2. *Push – Event based,* yaitu pesan yang diaktivasi oleh aplikasi berdasrkan kejadian yang berlangsung.
3. *Push – Schedule,* yaitu pesan yang diaktivasi oleh aplikasi berdasarkan waktu yang telah terjadwal.
4. *Push – Personal Profil,* yaitu pesan yang diaktivasi oleh aplikasi berdasrkan *profil* dan *preference* dari pengguna.

II.2.3 Definisi SMS

Short Message Service menurut [8] *network element and architeture* adalah sebuah mekanisme penyampaian pesan pendek dalam jaringan bergerak. SMS saat ini menjadi sebuah fitur mendasar dari setiap dari telepon seluler. Fitur SMS baru berjalan ketika terdapat operator telepon seluler (*operator provider*) yang menyediakan layanan ini. SMS memungkinkan dua orang yang memiliki telepon seluler dapat saling mengirimkan pesan teks atau satu sama lain.

Dengan perkembangan teknologi informasi saat ini, pengguna maupun pengembangan fitur-fitur SMS sendiri telah menjadi sangat beragam. Mulai dari keperluan kuis, berita umum, promosi produk, iklan dan lain sebagainya. Hal ini dapat terjadi karena data maupun proses pengolahan SMS telah dapat diintegerasikan dengan dengan komputer maupun internet, sehingga dapat diprogram untuk berbagai keperluan.

Teknologi yang mendukung SMS antara lain adalah GSM, TDMA dan CDMA dengan didukung oleh ketiga teknologi ini, SMS telah menjadi layanan data bergerak yang bersifat universal. Protocol yang bekerja pada SMS ini lebih dikenal dengan nama *Protocol Data Unit* (PDU).

II.3 Algoritma DES

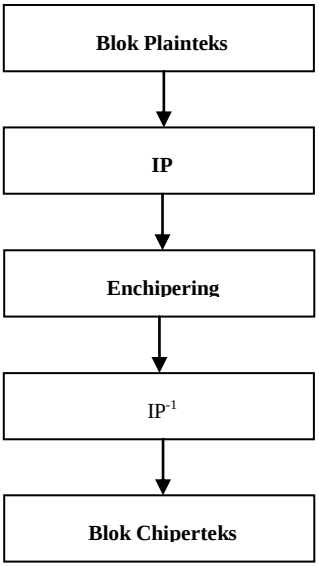
DES (*Data Encryption Standard*) adalah algoritma *chiper* blok yang populer karena dijadikan standard algoritma enkripsi kunci simetri, meskipun saat ini standard tersebut telah digantikan dengan algoritma yang baru, *AES*, karena DES sudah diaggap tidak aman lagi. Sebenernya DES adalah nama standard enkripsi simetri, nama algoritma enkripsinya sendiri adalah DEA (*Data Encryption Algorithm*) namun nama DES lebih populer daripada DEA. Algoritma DES dikembangkan di IBM dibawah kepemimpinan W.L. Tuchman pada tahun 1972. Algorima ini didasarkan pada algoritma *Lucifer* yang dibuat oleh *Horst Feistel*. Algoritma ini telah disetujui oleh *National Burea of Standard* (NBS) setelah penilaian kekuatannya oleh *National Security Agency* (NSA) Amerika Serikat.

DES termasuk ke dalam sistem kriptografi simetri dan tergolong jenis *Chiper* blok. DES beroprasi pada ukuran blok 64 bit. DES mengenkripsikan 64 bit

plainteks menjadi 64 bit chiperteks dengan menggunakan 56 bit kunci internal (*Internal Key*) atau upa kunci (*Subkey*). Kunci internal dibangkitkan dari kunci eksternal (*eksternal key*) yang panjangnya 64 bit.

Dalam prakteknya proses pembalikan (proses dekripsi) ini diimplementasikan dengan membelikan urutan sub kunci yang digunakan dalam proses enkripsi, selebihnya algoritma enkripsi dan dekripsi adalah sama. Algoritma enkripsi DES bekerja dengan mengolah blok data 8 byte (64 bit) dengan blok kunci 64 bit. Proses penyandian dalam DES diawali dengan fungsi pegacakan bit yang dinamai IP (*initial Permulatation*) kemudian fungsi inti DES yang diulang sebanyak 16 kali dan terakhir ditutup dengan fungsi pengacakan bit lain yang dikenal dengan nama IP^{-1} (*Inverse initial Permutation*). Pada sisi lain algoritma penjadwalan sub akan menghasilkan 16 sub kunci secara berurutan dari parameter kunci yang diberikan untuk digunakan pada pada setiap putaran fungsi inti DES. Sub kunci pertama untuk putaran pertama, sub kunci kedua untuk putaran kedua dan seterusnya hingga putaran ke 16.

Perlu diingat, kendaptipun slot kunci yang disediakan digunakan berukuran 64 bit, ternyata pada faktanya ukuran kunci yang digunakan hanya sebanyak 56 bit saja, karena bit paling signifikan (MSB) dari setiap bit diabaikan.[5]



Gambar II.2 Skema Global Algoritma DES
(sumber : MocoCorner.com : 2014)

Skema global dari algoritma DES adalah sebagai berikut :

1. Blok plainteks dipermutasi dengan matriks permutasi awal (*Initial Permutation* atau IP).
2. Hasil permutasi awal kemudian di *enciphering* sebanyak 16 kaH (16 putaran). Setiap putaran menggunakan kunci internal yang berbeda.
3. Hasil *enciphering* kemudian dipermutasi dengan matriks permutasi balikan (*Invers Initial Permutation* atau IP^{-1}) menjadi blok cipherteks.

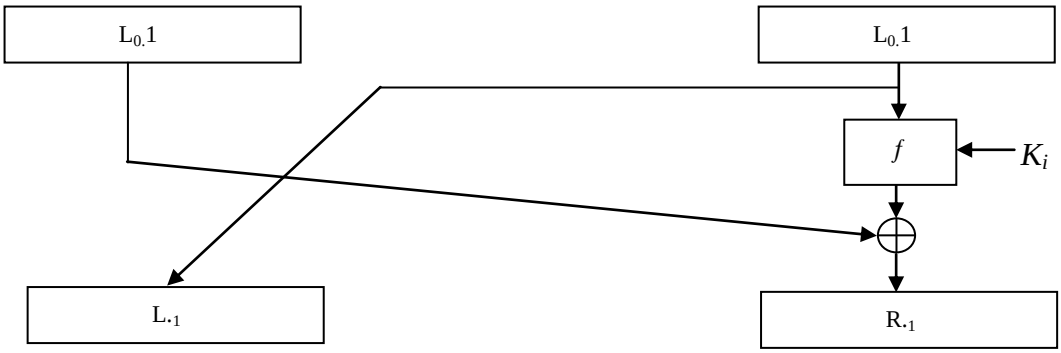
Didalam proses *enciphering*, blok plainteks terbagi menjadi dua bagian, kiri (L) dan kanan (R), yang masing-masing panjangnya 32 bit. Kedua bagian ini masuk kedalam 16 putaran DES. Pada setiap putaran i, blok R merupakan masukan untuk fungsi transformasi yang disebut F. Pada fungsi F, blok R dikombinasikan dengan kunci internal K. Keluaran dari fungsi F di XOR kan dengan blok L untuk mendapatkan blok R yang baru. Sedangkan blok yang baru

langsung diambil dari blok R sebelumnya. Ini adalah satu putaran DES, secara watermatis satu putaran DES dinyatakan sebagai

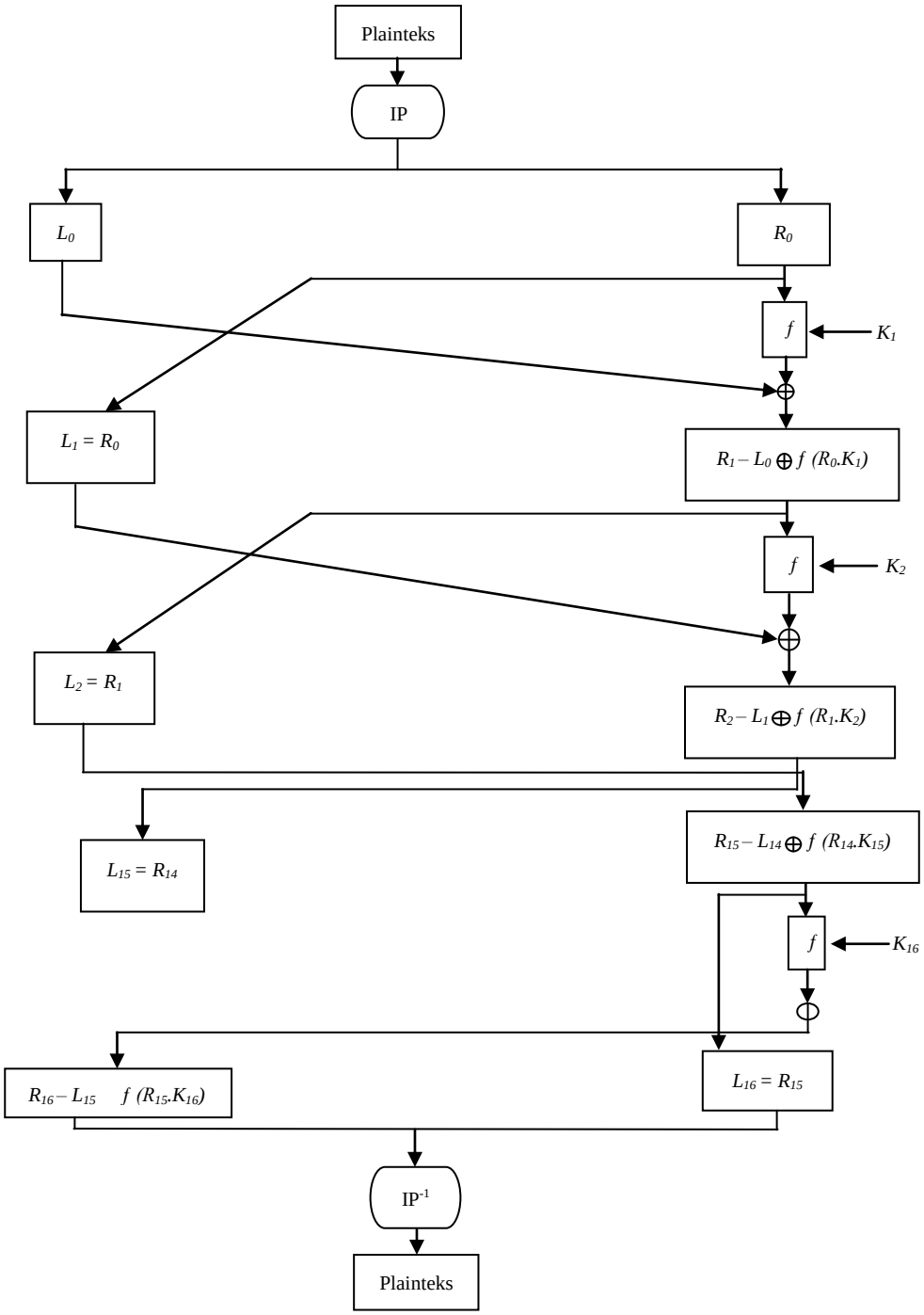
$L_i = R_{i-1}$

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

Satu putaran DES merupakan model jaringan *Feistel*. Perlu dicatat dari gambar II.2 bahwa ika (L,6, R,6) merupakan pra-cipherteks (*pre-chipertext*) dari *enciphering* ini. *Chipertext* yang sebenarnya diperoleh dengan melakukan permutasi awal balikan, IP^{-1}, terhadap blok pra-cipherteks.

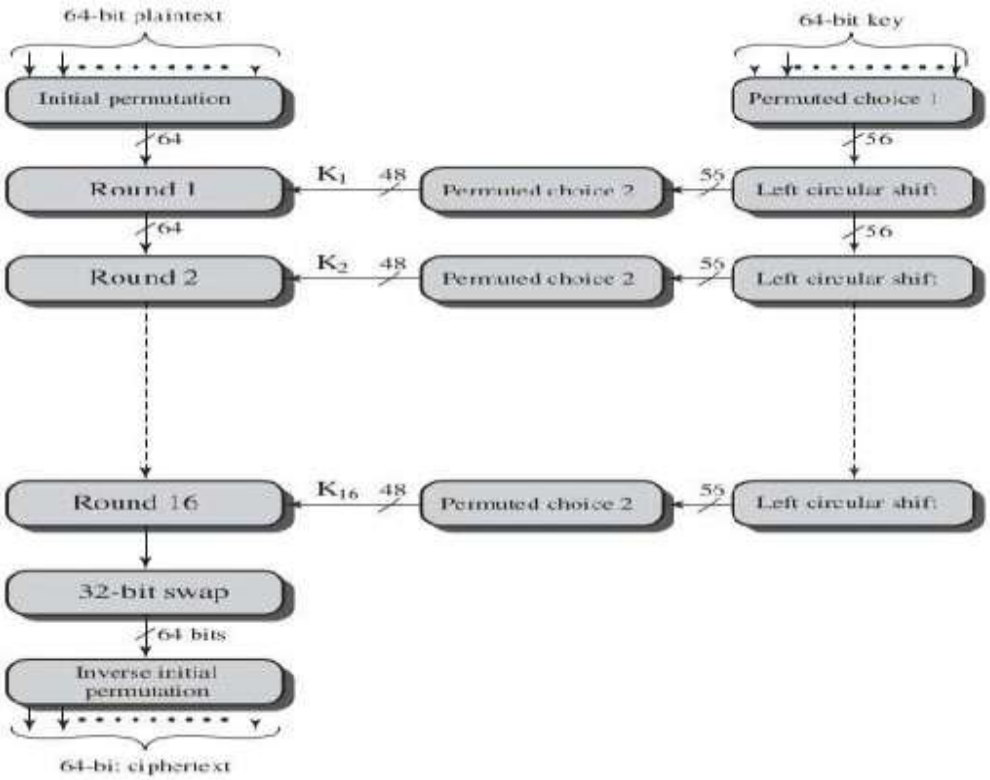


Gambar II.3 Jaringan Feistel Untuk Satu Putaran DES
(Sumber : Jimmy Jarred : 2013 : 23)



Gambar II.4 Algoritma Enkripsi Dengan DES Permutasi Awal
(Sumber : Jimmy Jarred : 2013 : 24)

Algoritma DES memerlukan sebuah kunci yang panjang bloknya 64 bit di setiap blok DES digunakan untuk mengamankan data pada perangkat lunak dan keras dinegara tersebut. Berikut desain input-output algoritma DES



Gambar II.5 Input - Output Algoritma DES
(Sumber : MocoConer.com : 2014)

Dapat dilihat bahwa ada dua input untuk fungsi enkripsi, yaitu plaintext dengan panjang 64-bit dan kunci dengan panjang 56-bit. Untuk mengenkripsi data dengan menggunakan algoritma DES, dimulai dengan membagi bit dari teks tersebut kedalam blok-blok dengan ukuran blok sebesar 64-bit, yang kemudian disebut blok plaintext.

Sebelum putaran pertama, terhadap blok plainteks dilakukan permutasi awal (*initial-permutation* atau IP). Tujuan permutasi awal adalah mengacak plainteks sehingga urutan bit-bit di dalamnya berubah. Pengecekan dilakukan dengan menggunakan matriks permutasi awal berikut ini :

58	50	42	34	26	18	10	2	60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6	64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1	59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5	63	55	47	39	31	23	15	7

Cara membaca tabel tau matriks : dua *entry* ujung kiri ke atas (58 dan 50) artinya :

"pindahkan bit ke-58 ke posisi bit 1"

"pindahkan bit ke-50 ke posisi bit 2", dst

II.3.1 Pembangkitan Kunci Internal

Karena ada 16 putaran, maka dibutuhkan kunci internal sebanyak 16 buah, yaitu K_1, K_Z, ..., K_{16}. Kunci-kunci internal ini dapat dibangkitkan sebelum proses enkripsi atau bersamaan dengan proses enkripsi. Kunci internal dibangkitkan dari kunci eksternal yang diberikan oleh pengguna. Kunci eksternal panjangnya 64 bit atau 8 karakter. Misalkan kunci eksternal yang tersusun dari 64 bit adalah *K*. Kunci eksternal ini menjadi masukan untuk permutasi dengan menggunakan matriks permutasi kompresi PC-1 sbagai berikut :

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

Dalam permutasi ini, tiap bit kedalapan (*parity bit*) dari delapan *byte* kunci diabaikan. Hasil permutasinya adalah sepanjang 56 bit, sehingga dapat dikatakan panjang kunci DES adalah 56 bit. Selanjutnya, 56 bit ini dibagi menjadi 2 bagian, kiri dan kanan yang masing-masing panjangnya 28 bit, yang masing-masing disimpan di dalam Co dan Do :

Co: berisi bit-bit dari *K* pada posisi

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

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

Do: berisi bit-bit dari *K* pada posisi

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

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

Selanjutnya, kedua bagian digeser kekiri (*Left Shif*) sepanjang satu atau dua bit bergantung pada putaran.

II.4 Android

Android adalah sebuah sistem operasi mobile yang berbasiskan pada versi modifikasi dari linux. Pertama kali system operasi ini dikembangkan oleh perusahaan Andrid.Inc. nama perusahaan inilah yang pada akhirnya digunakan sebagai nama proyek system operasi mobile tersebut, yaitu system operasi android.

Pada tahun 2005, sebagian dari strategi untuk memasuki pasar mobile, google membeli android dan mengambil alih proses pegembangannya sekaligus team develover andriod. Google menginginkan android untuk menjadi sistem operasi *Open Source* dan gratis, kebanyakan code android dirilis dibawah lisensi *Open Source Aphace* yang berarti setiap orang bebas untuk menggunakan dan mengunduh *Source Code* android secara penuh.

Android telah dikembangkan dan di *update* beberapa kali sejak rilis pertamanya. Tabel II.1 dibawah ini memperlihatkan versi android semenjak pertama kali dirilis. [3]

Tabel II.1 Versi Android (Sumber : Wahana Komputer : 2013 : 2-3)

Versi Android	Tanggal Rilis	Nama Kode
1.1	9 Februari 2009	-
1.5	30 April 2009	Cupcake
1.6	15 September 2009	Donut
2.0 / 2.1	26 Oktober 2009	Eclair
2.2	20 Mei 2010	Froyo
2.3	6 Desember 2010	Gingerbread
3.0	22 Februari 2011	Honeycomb
4.0	19 Oktober 2011	Ice Cream Sandwich
4.1	27 Juni 2012	Jelly Bean

II.5 Eclipse

Salah satu tool utama dalam proses pembangunan aplikasi android adalah *Eclipse*, yaitu sebuah IDE (*Integrate Development Environment*) untuk mengembangkan perangkat lunak dan dapat dijalankan disemua *platform* (*Platform Independent*). Berikut adalah sifat dari *Eclipse* :

1. Multi-platform : target sistem operasi *Eclipse* adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.
2. Multi-language : *Eclipse* dikembangkan dengan bahasa pemrograman *Java*, namun *Eclipse* mendukung pengembangan aplikasi berbasis pemrograman lainnya, seperti C/C++, Cobol, Python, Perl, PHP dan sebagainya.
3. Multi-role : selain sebagai IDE untuk pengembangan aplikasi *Eclipse* pun biasa digunakan untuk aktifitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, tes perang kat lunak, pengembangan web dan sebagainya.

Sejak versi 3.0 pada dasarnya *Eclipse* merupakan kernel yang sangat tergantung kepada *Plug-in* untuk melakukan aksinya. Fitur-fitur didalam *Eclipse* sebenarnya adalah fungsi dari *Plug-in* yang sudah diinstall. Ini merupakan paradigma dari *Eclipse* yang dinamakan *Rich Client Platform* (RCP). Berikut adalah komponen yang membentuk RCP :

1. *Core Platform*.
2. OSGi
3. SWT (*Standart Widget Toolkit*)
4. JFAce

Eclipse selalu dilengkapi dengan JDT (*Java Development Tools*), yaitu sebuah *Plug-in* yang membuat *Eclipse* dapat dipakai untuk mengembangkan program *Java* serta ada juga PDE (*Plugin Develoment Environment*) yang dapat dipakai untuk membuat *Plug-in* baru. [3]

II.5.1 Aplikasi *Native*

Aplikasi *native* adalah aplikasi yang secara khusus ditujukan untuk *platfrorm* mobile tertentu dan menggunakan bahasa pemrograman serta perangkat lunak pengembangan sesuai dengan *platform* tersebut. Sebagai contoh, aplikasi *native* android ditulis menggunakan bahasa bahasa pemrograman *Java* dan *Tool Eclipse*, sementara aplikasi iOS/iPhone dibuat dengan menggunakan bahasa *Objective-C* dan *tool Xcode.*

a) Kelibihan aplikasi *native*

1. Performa yang sangat baik karena ditulis secara *native* untuk *platform* spesifik.

2. Mampu mengakses semua fitur perangkat keras *smartphone*, seperti *info device, accelerator*, kamera, kompas, file dan lain sebagainya.
3. Menghasilkan antarmuka *look and feel* yang alami dengan sangat baik.

b) Kekurangan aplikasi *native*

1. Pengembangan yang tidak mudah karena menggunakan lingkungan bahasa dan API (*Application Programming Interface*) spesifik.
2. Aplikasi hanya berjalan pada *platform* yang sudah dispesifikasikan diawali pengembangan. Apabila ingin dkembangkan di *platform* lain maka harus ditulis ulang dengan *tool* pengembangan yang sesuai. [5]

II.5.2 Tool Pengembangan Android

Untuk mengembangkan sistem operasi android dapat menggunakan Mac, Windows atau PC, Linux, semua tool yang dibutuhkan adalah gratis dan dapat di download dari web.

1. *Java* JDK : Android berjalan dengan menggunakan *resource* dari *Java SE Development Kit* (JDK).
2. Android SDK : Android SDK berikan *Debugger*, *library*, dokumentasi, kode conto dan tutorial. Android SDK dapat di download dari alamat : http://developer.android/sdk.index.html.
3. *Android Development Tools* (ADT) : *plug-in Android development Tools* (ADT) untuk mendukung pembuatan dan proses *debugging* dari aplikasi android yang sedang dibuat.

II.6 Use Case Diagram

Dalam bahasa pemodelan ini, penulis menggunakan 1 (satu) bauh hactor yaitu user. User pada aplikasi ini adalah seseorang yang nantinya akan menjalankan aplikasi *MySMSCrypt* ini.

Tabel II.2 Skenario Use Case Proses Kirim dan Enkrip

Email Use Case : 1. Proses kirim dan enkrip email
Primary actor : 1. User
Goal : 1. User bisa mengirim dan mengenkripsikan SMS
Precondition : 1. User mengisikan konfigurasi dengan benar 2. User mengisikan nomor telepon tujuan dan mengisi key enkrip dengan benar.
Triger : 1. User ingin melakukan pengiriman dan enkripsi email
Scenario : 1. User membuka menu kirim SMS 2. User mengisi nomor telepon yang dituju, isi pesan dan tekan tombol enkrip, mengisikan key untuk mengenkripsi pesan. 3. User mengeksekusi fungsi aplikasi enkrip dan kirim SMS.
Alternate Flow : 1. User keluar dari aplikasi

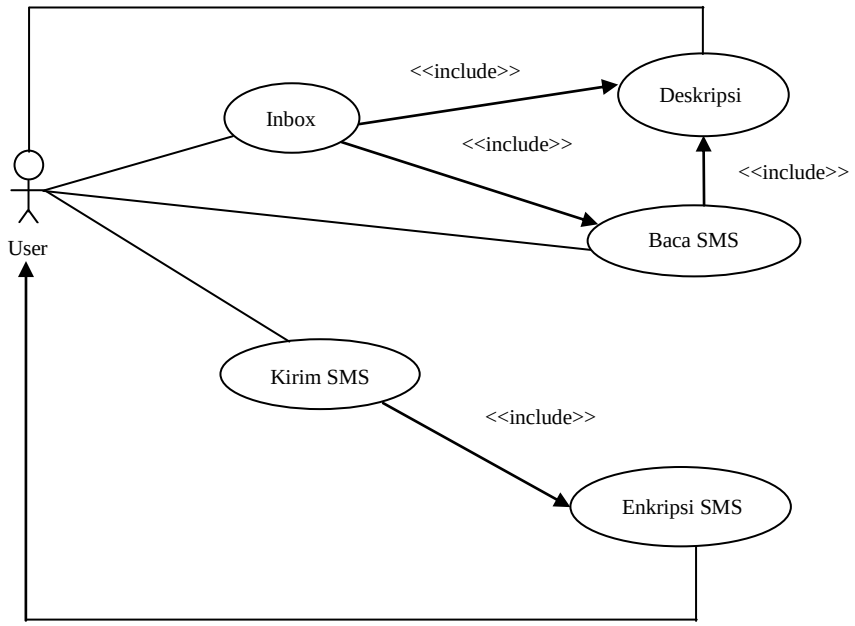
Priority : 1. Moderate priority
Frequency of use : 1. Frequent

Tabel II.3 Skenario Use Case Proses Inbox Dekripsi

Use Case : 1. Proses Inbox Dekripsi.
Primary Actor : 1. User
Goal : 1. User terhubung Provider dan dapat melihat daftar SMS masuk, mendekripsi
Precondition : 1. User mengisikan konfigurasi dengan benar. 2. User mengisikan key untuk dekripsi SMS dengan benar
Triger : 1. User ingin masuk kedalam kotak masuk SMS pada provider
Scenario : 1. User membuka menu inbox 2. User memilih pesan yang akan didekripsi dan memasukan kunci
Alternate Flow : 1. User keluar dari aplikasi

Priority : 1. Moderate priority
Frequency of use : 1. Frequent

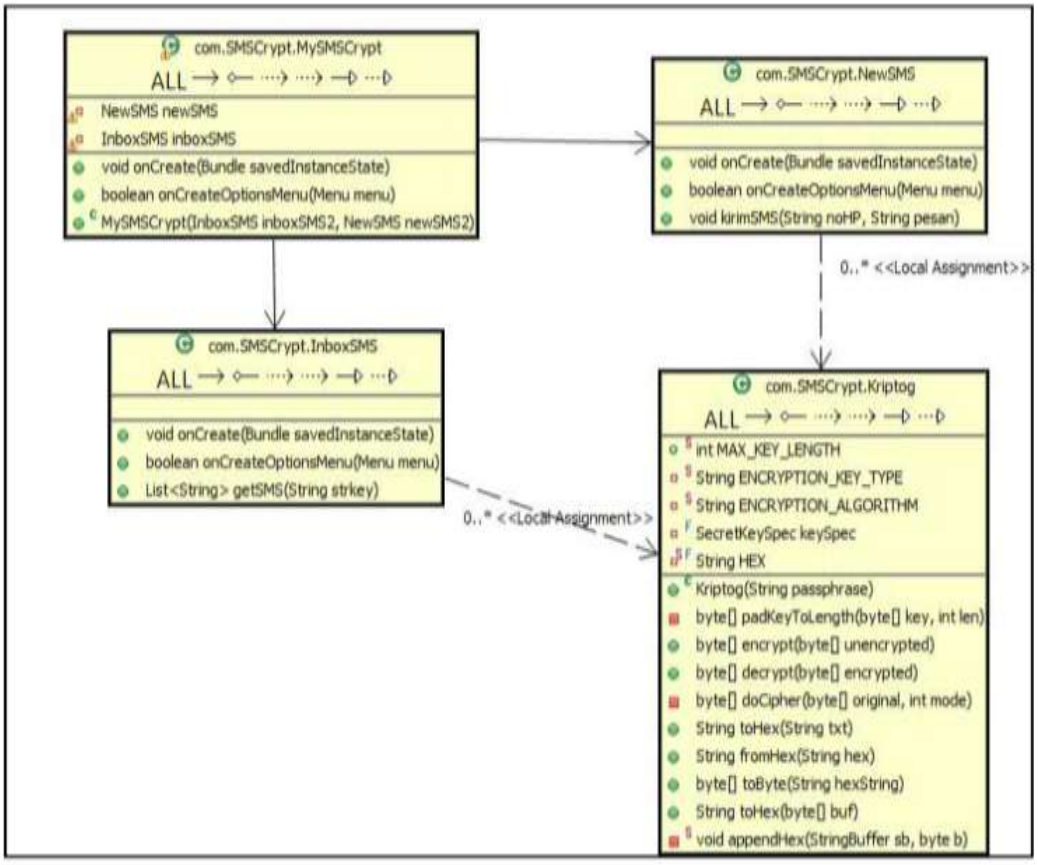
Dibawah ini merupakan pemodelan use case yang penulis pakai pada pembuatan aplikasi ini.



Gambar II.6 Use Case Aplikasi MySMSCript
(Sumber : Jurnal Agus Abdullah : 2014)

II.6.1 Class Diagram

Pada class diagram, penulis menggunakan 4 macam kelas yaitu : MySMSCrypt, NewSMS, InboxSMS dan Kripto. Kelas-kelas tersebut saling berhubungan dan mempunyai ke terkaitan. Dibawah ini merupakan gambar dan penjelasan yang dimaksud :



Gambar II.7 Class Diagram Aplikasi MySMSCrypt
(Sumber : Jurnal Agus Abdullah : 2014)

1. MySMSCrypt : Class yang berisikan tentang menu-menu yang akan di jalankan pada aplikasi.
2. NewSMS : Class yang dijalankan untuk pemprosesan pengirim dan pengenkripsian SMS.
3. InboxSMS : Class untuk menampilkan isi dan pendekripsian SMS
4. Kripto : Class public untuk enkripsi dan deskripsi