BAB II

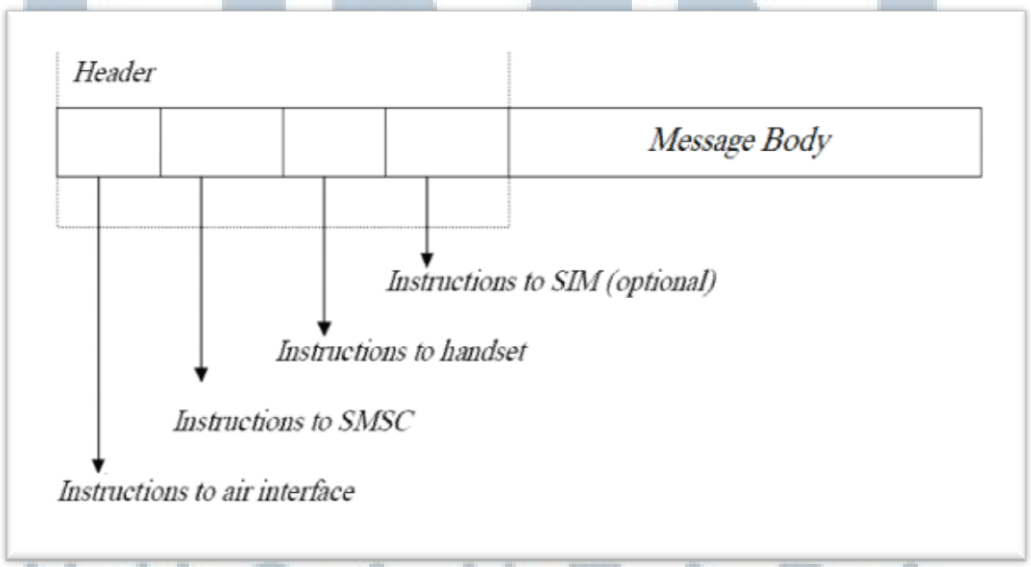
TINJAUAN PUSTAKA

2.1 Short Message Service

SMS merupakan salah satu layanan pesan teks yang dikembangkan dan distandarisasi oleh suatu badan yang bernama *ETSI*(*European Telecomunication Standards Institute*) yang dapat mengirim pesan-pesan teks dengan panjang sampai dengan 160 karakter melalui jaringan *GSM*(Wahana Komputer, 2005).

Layanan *SMS* juga akan mengirimkan *SMS* sampai ke tujuan meskipun perangkat tujuan sedang tidak aktif atau sedang berada di luar jangkauan layanan. *SMS* terlebih dahulu akan disimpan pada *Short Message Service Center* sebelum kemudian dikirim lagi jika sudah ada tanda keaktifan dari perangkat yang dituju (Clements, 2003).

Struktur pesan pada sebuah paket *SMS* dapat dilihat pada gambar di bawah ini.



Gambar 2.1 Struktur *SMS*

(Sumber: http://developers.sun.com/techtopics/mobility/midp/articles/sms/)

Pada gambar 2.1 terlihat bahwa paket *sms* terdiri dari *header* dan *body*. *Header* pesan terdiri dari instruksi-instruksi kepada komponen-komponen yang bekerja dalam jaringan *sms*. Pada instruksi-instruksi tersebut, terdapat informasi yang diperlukan selama pengiriman pesan seperti informasi validitas pesan, dan informasi informasi lainnya. Pada bagian *message body*, terdapat isi dari pengirim pesan yang akan dikirimkan (Clements, 2003).

2.2 Kriptografi

Kriptografi berasal dari bahasa Yunani yaitu *kryptos* yang artinya "yang tersembunyi" dan *graphein* yang artinya "tulisan". Jadi kriptografi adalah seni dan ilmu untuk menjaga keamanan data dan ahlinya disebut sebagai *cryptographer*. *Cryptanalst* merupakan orang yang melakukan cryptanalysis, yaitu seni dan ilmu untuk membuka ciphertext menjadi plaintext tanpa melalui cara yang seharusnya. Data yang dapat dibaca disebut *plaintext* dan teknik untuk membuat data tersebut menjadi tidak dapat dibaca disebut enkripsi. Data hasil dari enkripsi disebut *ciphertext*, dan proses untuk mengembalikan *ciphertext* menjadi *plaintext* disebut dekripsi atau *decipher* (Ariyus, 2008).

2.2.1 Algoritma Kriptografi

Perkembangan algoritma kriptografi dapat dibagi menjadi 2 yaitu:

- Kriptografi Klasik
- Kriptografi Modern

2.2.2 Kriptografi Klasik

Algoritma ini merupakan algoritma kriptografi yang biasa digunakan orang sejak berabad-abad yang lalu (Yusuf,2004). Dua teknik dasar yang biasa digunakan, yaitu:

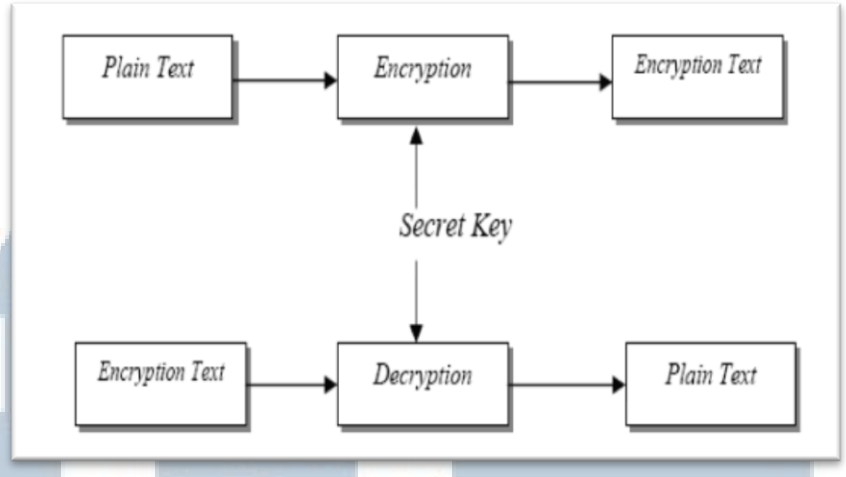
a. Teknik Subsitusi: penggantian setiap karakter *plaintext* dengan karakter lain.

b. Teknik Transposisi: Teknik ini menggunakan permutasi karakter.

Salah satu kriptografi yang masuk dalam kategori kriptografi klasik yaitu ROT13 *cipher.*

2.2.3 Kriptografi Modern

Algoritma modern selain memfokuskan diri pada tingkat kesulitan algoritma juga pada kunci yang digunakan. Macam-macam algoritma menurut kuncinya adalah algoritma simetris dan algoritma asimetris. Algoritma simetris disebut juga sebagai algoritma konvensional, yaitu algoritma yang menggunakan kunci yang sama untuk proses enkripsi dan deskripsinya. Keamanan algoritma simetris tergantung pada kuncinya. Algoritma simetris sering juga disebut algoritma kunci rahasia, algoritma kunci tunggal atau algoritma satu kunci (Yusuf, 2004).



Gambar 2.2 Kriptografi Simetris

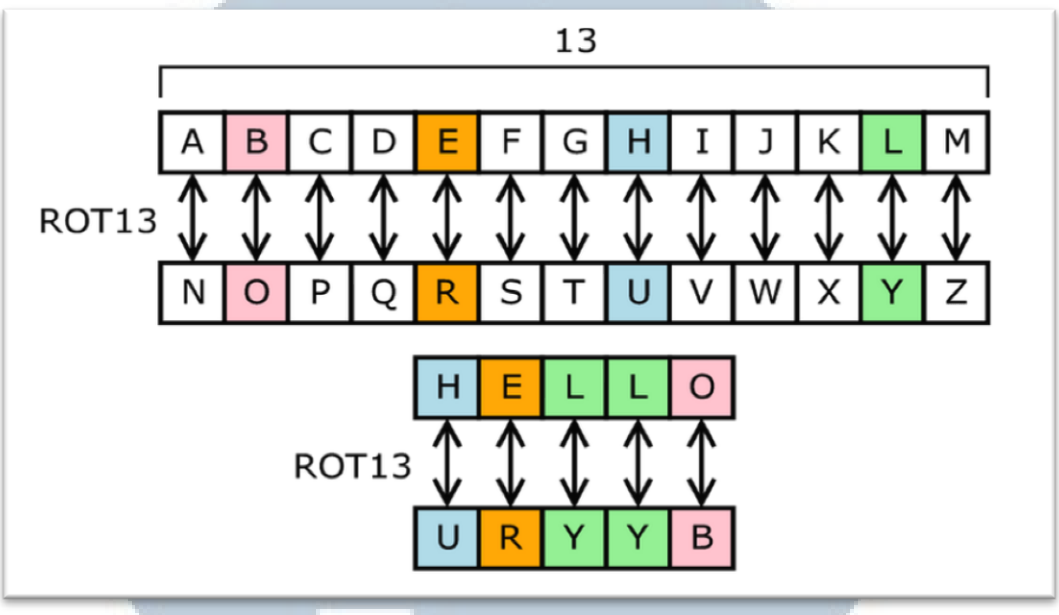
(Sumber:http://www.erdisusanto.com/2012/10/konsep-dasar-kriptografi-simetris-dan.html)

Salah satu algoritma yang masuk dalam kategori algoritma simetris yaitu algoritma *RC6*.

2.3 ROT 13

ROT13 sendiri merupakan bagian pengembangan dari algoritma *Caesar Cipher*. Pada sistem ini sebuah huruf digantikan dengan huruf yang letaknya 13 posisi darinya. Sebagai contoh, huruf “A” digantikan dengan huruf “N”, huruf “B” digantikan dengan huruf “O”, dan seterusnya (Nugroho, 2012). Dibawah merupakan contoh gambar dari ROT13 *cipher*: Contoh *pseudocode* dari ROT13 cipher adalah sebagai berikut:

```
if (letter > 'm') { letter -= 13; }
else { letter += 13; }
if (letter > 'M') { letter -= 13; }
else { letter += 13; }
```



Gambar 2.3 ROT13 Cipher

Sumber: http://jacklyons.me/bonfire-caesars-cipher/)

2.4 Algoritma RC6

Algoritma *RC6* merupakan salah satu kandidat *Advanced Encryption Standard (AES)* yang diajukan oleh *RSA* Laboratoriest kepada *NIST*(*National Institute of Standards and Technology*). Dirancang oleh Ronald L Rivest, M.J.B. Robshaw, R. Sidney dan Y.L. Yin, algoritma ini merupakan pengembangan dari algoritma sebelumnya yaitu *RC5* dan telah memenuhi semua kriteria yang diajukan oleh *NIST*.

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 negatif yang menunjukan banyaknya iterasi selama proses enkripsi, dan b

merupakan ukuran kunci enkripsi dalam *byte*. Nilai parameter w = 32, r = 20 dan b bervariasi antara 16, 24, dan 32 *byte* (Abdurohman, 2002).

RC6-w/r/b memecah blok 128 bit menjadi 4 buah blok 32 bit, dan mengikuti enam aturan operasi dasar sebagai berikut (Prayudi, 2005) :

A + B Operasi penjumlahan bilangan integer.

A – B Operasi pengurangan bilangan integer.

A XOR 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 kanani sebanyak variabel kedua (B)

RC6 memecah *block* 128 bit menjadi 4 buah *block* 32 bit, maka algoritma ini bekerja dengan empat 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* pada sisi kiri (Abdurohmman, 2002). Berikut ini adalah algoritma enkripsi RC6.

```
for i = 1 to 20 do
{
      t = ( B x ( 2B + 1 ) )
      <<< 5
      u = ( D x ( 2D + 1 ) )
      <<< 5
      A = ( ( A XOR t ) <<< u )
      + S[ 2i ]
      C = ( ( C XOR u ) <<< t )
      + S[ 2i + 1 ]
      (A, B, C, D) = (B, C, D, A)
}
A = A + S[ 42 ]
C = C + S[ 43 ]
```

Gambar 2.4 Algoritma Enkripsi RC6

(Sumber: https://spyn3t.wordpress.com/2007/12/24/rc6-block-cipher/)

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. Berikut tabel sub kunci dari algoritma *RC6*:

Tabel 2.1 Sub Kunci Algoritma RC6

Sbox	Hexa	Decimal	Binary
0	B7E15163	3,084,996,963	1011 0111 1110 0001 0101 0001 0110 0011
1	5618CB1C	1,444,465,436	0101 0110 0001 1000 1100 1011 0001 1100
2	F45044D5	4,098,901,205	1111 0100 0101 0000 0100 0100 1101 0101
3	9287BE8E	2,458,369,678	1001 0010 1000 0111 1011 1110 1000 1110
4	30BF3847	817,838,151	0011 0000 1011 1111 0011 1000 0100 0111
5	CEF6B200	3,472,273,920	1100 1110 1111 0110 1011 0010 0000 0000
6	6D2E2BB9	1,831,742,393	0110 1101 0010 1110 0010 1011 1011 1001
7	0B65A572	191,210,866	0000 1011 0110 0101 1010 0101 1000 0010
8	A99D1F2B	2,845,646,635	1010 1001 1001 1101 0001 1111 0010 1011
9	47D498E4	1,205,115,108	0100 0111 1101 0100 1001 1000 1110 0100
10	E60C129D	3,859,550,877	1110 0110 0000 1100 0001 0010 1001 1101
11	84438C56	2,219,019,350	1000 0100 0100 0011 1000 1100 0101 0110
12	227B060F	578,487,823	0010 0010 0111 1011 0000 0110 0000 1111
13	C0B27FC8	3,232,923,592	1100 0000 1011 0010 0111 1111 1100 1000
14	5EE9F981	1,592,392,065	0101 1110 1110 1001 1111 1001 1000 0001
15	FD21733A	4,246,827,834	1111 1101 0010 0001 0111 0011 0011 1010
16	9B58ECF3	2,606,296,307	1001 1011 0101 1000 1110 1100 1111 0011
17	399066AC	965,764,780	0011 1001 1001 0000 0110 0110 1010 1100
18	D7C7E065	3,620,200,549	1101 0111 1100 0111 1110 0000 0110 0101
19	75FF5A1E	1,979,669,022	0111 0101 1111 1111 0101 1010 0001 1110
20	1436D3D7	339,137,495	0001 0100 0011 0110 1101 0011 1101 0111
21	B26E4D90	2,993,573,264	1011 0010 0110 0110 0100 1101 1001 0000
22	50A5C749	1,353,041,737	0101 0000 1010 0101 1100 0111 0100 1001
23	EEDD4102	4,007,477,506	1110 1110 1101 1101 0100 0001 0000 0010
24	8D14BABB	2,366,945,979	1000 1101 0001 0100 1011 1010 1011 1011
25	2B4C3474	726,414,452	0010 1011 0100 1100 0011 0100 0111 0100
26	C983AE2D	3,380,850,221	1100 1001 1000 0011 1010 1110 0010 1101
27	67BB27E6	1,740,318,694	0110 0111 1011 1011 0010 0111 1110 0110
28	05F2A19F	99,787,167	0000 0101 1111 0010 1010 0001 1001 1111
29	A42A1B58	2,754,222,936	1010 0100 0010 1010 0001 1011 0101 1000
30	42619511	1,113,691,409	0100 0010 0110 0001 1001 0101 0001 0001
31	E0990ECA	3,768,127,178	1110 0000 1001 1001 0000 1110 1100 1010
32	7ED08883	2,127,595,651	0111 1110 1101 0000 1000 1000 1000 0011
33	1D08023C	487,064,124	0001 1101 0000 1000 0000 0010 0011 1100

Tabel 2.1 Sub Kunci Algoritma RC6 (Lanjutan)

Sbox	Hexa	Decimal	Binary
34	BB3F7BF5	3,141,499,893	1011 1011 0011 1111 0111 1011 1111 0101
35	5976F5AE	1,500,968,366	0101 1001 0111 0110 1111 0101 1010 1110
36	F7AE6F67	4,155,404,135	1111 0111 1010 1110 0110 1111 0110 0111
37	95E5E920	2,514,872,608	1001 0101 1110 0101 1110 1001 0010 0000
38	341D62D9	874,341,081	0011 0100 0001 1101 0110 0010 1101 1001
39	D254DC92	3,528,776,850	1101 0010 0101 0100 1101 1100 1001 0010
40	708C564B	1,888,245,323	0111 0000 1000 1100 0101 0110 0100 1011
41	0EC3D004	247,713,796	0000 1110 1100 0011 1101 0000 0000 0100
42	ACFB49BD	2,902,149,565	1010 1100 1111 1011 0100 1001 1011 1101
43	4B32C376	1,261,618,038	0100 1011 0011 0010 1100 0011 0111 0110

(Sumber: http://catatanrimbun.blogspot.co.id/2012/12/algoritma-rivest-code-6-rc6.html)

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 di mana 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 ke kiri 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 ke kiri. Begitu pula dengan nilai u, juga digunakan sebagai acuan bagi nilai A untuk melakukan proses pemutaran ke kiri.

Kemudian sub kunci S[2i] pada iterasi dijumlahkan dengan A, dan sub kunci S[2i+1] dijumlahkan dengan C. Keempat bagian dari *block* kemudian akan dipertukarkan dengan mengikuti aturan, bahwa nilai A ditempatkan pada D, nilai B ditempatkan pada A, nilai C ditempatkan pada B, dan nilai (asli) D ditempatkan pada C. Demikian iterasi tersebut akan terus berlangsung hingga 20 kali.

Proses dekripsi *ciphertext* pada algoritma *RC6* merupakan pembalikan dari proses enkripsi. Pada proses *whitening*, bila proses enkripsi menggunakan operasi penjumlahan, maka pada proses dekripsi menggunakan operasi pengurangan. Sub kunci yang digunakan pada proses *whitening* setelah iterasi terakhir diterapkan sebelum iterasi pertama, begitu juga sebaliknya sub kunci yang diterapkan pada proses *whitening* sebelum iterasi pertama digunakan pada *whitening* setelah iterasi terakhir.

Akibatnya, untuk melakukan dekripsi, hal yang harus dilakukan semata-mata hanyalah menerapkan algoritma yang sama dengan enkripsi, dengan tiap iterasi menggunakan sub kunci yang sama dengan yang digunakan pada saat enkripsi, hanya saja urutan sub kunci yang digunakan terbalik. Berikut ini adalah algoritma dekripsi RC6:

```
C = C - S[ 43 ]
A = A - S[ 42 ]
for i = 20 downto 1 do
{
    (A, B, C, D) = (D, A, B, C)
    u = ( D x ( 2D + 1 ) )
    <<< 5
    t = ( B x ( 2B + 1 ) )
    <<< 5
    C = ( ( C - S[ 2i + 1 ] ) >>> t ) XOR u
    A = ( ( A - S[ 2i ] ) >>> u) XOR t
}
D = D - S[ 1 ]
B = B - S[ 0 ]
```

Gambar 2.5 Algoritma Dekripsi RC6

(Sumber: https://spyn3t.wordpress.com/2007/12/24/rc6-block-cipher/)

Kemudian memasukkan sebuah kunci yang besarnya b *byte*, di mana 0≤b≤255. *byte* kunci ini kemudian ditempatkan dalam *array* c w-bit *words* L[0]…L[c-1]. *Byte* pertama kunci akan ditempatkan sebagai pada L[0], *byte* kedua pada L[1], dan seterusnya. (Catatan, bila b=0 maka c=1 dan L[0]=0). Masing-masing nilai kata w-bit akan dibangkitkan pada penambahan kunci *round* 2r+4 dan akan ditempatkan pada *array* S[0,..,2r+3]. Konstanta P32 = B7E15163 dan Q32 = 9E3779B9 (dalam satuan heksadesimal) adalah *magic constant* yang digunakan dalam penjadwalan kunci pada RC6. Nilai P32 diperoleh dari perluasan bilangan biner e-2, dimana e adalah sebuah fungsi logaritma. Sedangkan nilai Q32 diperoleh dari perluasan bilangan biner Ø-1,

dimana Ø dapat dikatakan sebagai *golden ratio.* Algoritma untuk pembangkitan kunci RC6 adalah sebagai berikut:

```
S[ 0 ] = 0xB7E15163
for i = 1 to 43 do S[i] = S[i-1]+ 0x9E3779B9
A = B = i = j = 0
for k = 1 to 132 do
{
      A = S[ i ] = ( S[ i ] + A + B)
      <<< 3
      B = L[ j ] = ( L[ j ] + A + B)
      <<< ( A + B )
      i = ( i + 1 ) mod 44
      j = ( j + 1 ) mod c
}
```

Gambar 2.6 Algoritma Pembangkit Sub Kunci

(Sumber: https://n3vrax.wordpress.com/2011/07/26/rc6-encryption-algorithm-in-java/)

2.5 Skala Likert

Skala Likert adalah merupakan sebuah metode yang dapat digunakan untuk mengukur sikap, pendapat dan presepsi seseorang atau kelompok orang tentang fenomena sosial (Sugiyono, 2012). Menurut Trochim (2006) skala Likert dapat lebih mudah diolah dan dipetakan ke dalam suatu kesimpulan.

Sikap dapat diukur dan intensitas suatu pengalaman adalah linear yaitu dituangkan di sebuah kontinum dari Sangat Setuju sampai Sangat Tidak Setuju dengan penilaian masing-masing pada setiap pilihan.(Likert, 1932)

1. Sangat tidak setuju
2. Tidak setuju
3. Tidak tahu / netral
4. Setuju
5. Sangat setuju

2.6 Black Box Testing

Pengujian dengan metode *black-box* merupakan pengujian yang menekankan pada fungsionalitas dari sebuah perangkat lunak tanpa harus mengetahui bagaimana struktur di dalam perangkat lunak tersebut. Sebuah perangkat lunak yang diuji menggunakan metode Black Box dikatakan berhasil jika fungsi-fungsi yang ada telah memenuhi spesifikasi kebutuhan yang telah dibuat sebelumnya (Khan, 2011). *Black Box Testing* cenderung untuk menemukan fungsi yang tidak benar atau tidak ada, kesalahan antarmuka, maupun kesalahan performansi. Berikut ini merupakan dua tahap dalam pengujuan pengujian sistem (Komarudin, 2013).

Tahap pertama yang dilakukan adalah pengujian alpha yaitu menguji secara langsung dengan cara uji coba data, yaitu dengan memasukan data yang sesuai atau benar dan juga dengan data yang salah. Sedangkan pengujian Beta merupakan pengujian sistem yang dilakukan secara objektif

yaitu pengujian langsung ke pengguna umum dengan menggunakan kuesioner untuk mengetahui pendapat dari responden terhadap sistem yang telah dibangun.