Implementasi Protokol Diffie-Hellman Dan Algoritma RC4 Untuk Keamanan Pesan SMS

Decky Hendarsyah dan Retantyo Wardoyo

Abstrack— SMS now becomes such a need for cellular phone users to communicate to other people. But the cellular phone users do not realize that the sent messages could be intercepted or changed by an unwanted party. Therefore it requires a security in sending an SMS message which is called cryptography. Given limited resources on cellular phone, then the implementation of symmetric cryptographic technique is suitable to meet the security needs of an SMS message. In symmetric cryptography, there is a symmetric key for encryption and decryption process. In order to secure exchange of symmetric keys in public channels is required of a protocol for key exchange.

This research implements RC4 symmetric cryptography to encrypt and decrypt messages, while for key exchange is using Diffie-Hellman protocol. In this research, there are modifications to the Diffie-Hellman protocol that is the calculation of the public key and symmetric key to include cellular phone number as authentication. Whereas on a modified RC4 is the key where there is a combination with cellular phone number as authentication and key randomization, and then there are also modifications to the pseudorandom byte generator, encryption and decryption of the RC4 algorithm. The system is constructed using the Java programming language in the platform Micro Edition (J2ME) based MIDP 2.0 and CLDC 1.0.

The research found that with the cellular phone number as authentication, key, encryption and decryption process automatically it is able to maintain confidentiality, data integrity, authentication and non-repudiation to the message.

Keywords— *Diffie-Hellman, Key exchange, RC4, SMS Secure, Symmetric Cryptography.*

I. PENDAHULUAN

Perkembangan teknologi dan komunikasi tumbuh dengan cepat, sehingga mengubah sistem komunikasi analog menjadi digital. Seiring dengan kemajuan sosial ekonomi masyarakat menuntut mobilitas yang semakin tinggi. Kemudian dilandasi dengan adanya kendala dalam pengembangan sistem komunikasi jaringan kabel akibat kondisi alam, maka dikembangkan teknologi seluler atau yang disebut dengan telepon seluler (*handphone*).

Telepon seluler menyediakan media layanan komunikasi yang beragam dan salah satunya adalah SMS (*Short Message Service*). SMS diartikan dalam bahasa Indonesia sebagai layanan pesan singkat. SMS merupakan suatu layanan yang memungkinkan pengguna telepon seluler dapat mengirim atau menerima pesan singkat kepada atau dari pengguna telepon seluler yang lain dengan cepat dan biaya yang relatif kecil.

Sewaktu proses pengiriman pesan pada SMS sangat besar kemungkinan celah keamanan, dimana pesan yang dikirim dapat disadap atau disusupi oleh pihak-pihak yang tidak diinginkan. Salah satu upaya pengamanan pesan SMS yang dapat dilakukan adalah dengan menggunakan kriptografi.

Kemajuan teknologi yang sangat pesat telah menciptakan telepon seluler yang memiliki memory cukup besar sehingga memungkinkan untuk melakukan implementasi kriptografi pada SMS. Akan tetapi telepon seluler secara umum tidak memiliki *processor* seperti *processor* pada PC (*Personal Computer*) yang mempunyai kecepatan tinggi, sehingga membatasi aplikasi yang mempunyai beban komputasi tinggi dan proses terlalu komplek pada telepon seluler.

Dengan keterbatasan sumber daya pada telepon seluler, maka implementasi teknik kriptografi simetrik sangat cocok untuk memenuhi kebutuhan keamanan pesan SMS, seperti algoritma RC4. Algoritma RC4 memiliki suatu kunci simetri untuk melakukan enkripsi dan dekripsi. Dalam pendistribusian kunci RC4 supaya lebih aman melalui jalur publik maka dibutuhkan suatu algoritma pertukaran kunci simetris yang tidak mempunyai trafik pesan yang tinggi yaitu menggunakan algoritma pertukaran kunci Diffie-Hellman. Algoritma ini sering juga disebut dengan protokol Diffie-Hellman atau Diffie-Hellman *key exchange* yang berguna untuk mempertukarkan kunci sesi (simetri). Tetapi protokol Diffie-Hellman dan algoritma RC4

D. Hendarsyah, Magister Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam,Universitas Gadjah Mada, Yogyakarta
e-mail : deckyh@mail.ugm.ac.id

R. Wardoyo, Magister Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam,Universitas Gadjah Mada, Yogyakarta
e-mail : rw@ugm.ac.id

rentan terhadap serangan, untuk itu perlu dilakukan modifikasi untuk mengatasi serangan tersebut.

Dengan demikian untuk menjaga keamanan pesan SMS agar tidak dapat dibaca oleh pihak yang tidak diinginkan maka diperlukan suatu sistem kriptografi. Dalam menemukan suatu sistem kriptografi yang mampu mengamankan pesan SMS dan dapat bekerja di telepon seluler yang memiliki sumber daya terbatas maka perlu dilakukan suatu penelitian mengenai implementasi modifikasi protokol Diffie-Hellman dan modifikasi algoritma RC4. Penelitian ini akan fokus pada proses pertukaran kunci, enkripsi dan dekripsi pesan SMS.

II. KONSEP KEAMANAN

Menurut Bishop [1], terdapat tiga komponen dasar sebagai pertimbangan dalam perancangan dan pembahasan sistem keamanan diantaranya adalah sebagai berikut:

- *Confidentiality* adalah penyembunyian informasi atau sumber daya yang berkaitan dengan pencegahan akan pengaksesan terhadap informasi atau sumber daya yang dilakukan oleh pihak yang tidak berhak.
- *Integrity* merupakan keandalan data atau sumber daya dan biasanya dirumuskan untuk mencegah perubahan yang tidak sah. Integritas mencakup integritas data (isi dari informasi) dan integritas asli (sumber data, sering disebut otentikasi). Dengan demikian *integrity* berkaitan dengan pencegahan modifikasi informasi yang dilakukan oleh pihak yang tidak berhak.
- *Availability* merupakan kemampuan untuk menggunakan informasi atau sumber daya yang diinginkan. *Availability* adalah aspek yang penting dalam mendesain sistem karena suatu sistem yang tidak memiliki *availability* sama buruknya dengan tidak ada sistem sama sekali. *Availability* dapat melakukan pencegahan akan penguasaan informasi atau sumber daya oleh pihak yang tidak berhak.

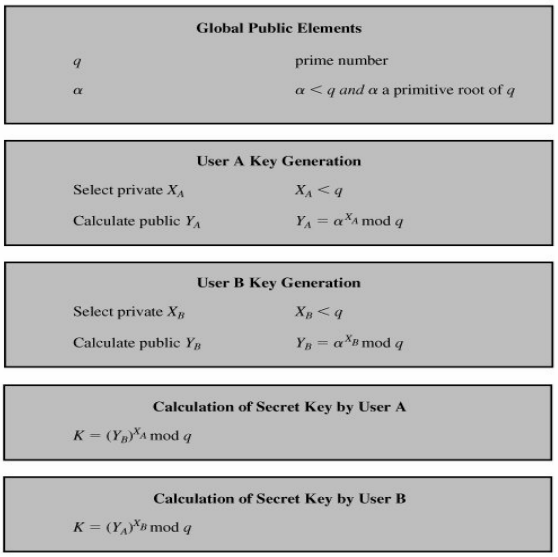
III. TUJUAN KRIPTOGRAFI

Dari beberapa tujuan keamanan informasi terdapat empat kerangka tujuan dari kriptografi diantaranya sebagai berikut [11] :

1. *Confidentiality* (kerahasiaan) adalah suatu layanan yang digunakan untuk menjaga isi informasi dari pihak yang tidak berwenang untuk memilikinya.
2. *Data integrity* (integritas data) adalah suatu layanan yang menjamin bahwa pesan masih asli atau belum pernah dimanipulasi selama pengiriman.
3. *Authentication* (otentikasi) adalah suatu layanan untuk mengidentifikasi kebenaran pihak-pihak yang berkomunikasi (*user authentication*) dan mengidentifikasi kebenaran sumber pesan (*data origin authentication*).
4. *Non-repudiation* (penyangkalan) adalah suatu layanan untuk mencegah entitas yang berkomunikasi melakukan penyangkalan, yaitu pengirim pesan menyangkal telah melakukan pengiriman pesan atau penerima pesan menyangkal telah menerima pesan.

IV. PROTOKOL DIFFIE-HELLMAN (DIFFIE-HELLMAN *KEY EXCHANGE*)

Stallings [17], menerangkan bahwa algoritma kunci publik diterbitkan pertama kali dalam makalah Diffie dan Hellman yang didefinisikan sebagai kriptografi kunci publik dan biasanya disebut sebagai Diffie-Hellman *Key Exchange* (pertukaran kunci) atau Protokol Diffie-Hellman. Tujuan dari algoritma ini adalah untuk memungkinkan dua pengguna saling bertukar kunci secara aman, kemudian dapat digunakan untuk enkripsi dan dekripsi pesan berikutnya. Algoritma itu sendiri terbatas pada pertukaran nilai-nilai rahasia seperti yang terlihat pada Gambar 1.



Gambar 1. Skema algoritma protokol Diffie-Hellman (Stallings, 2005)

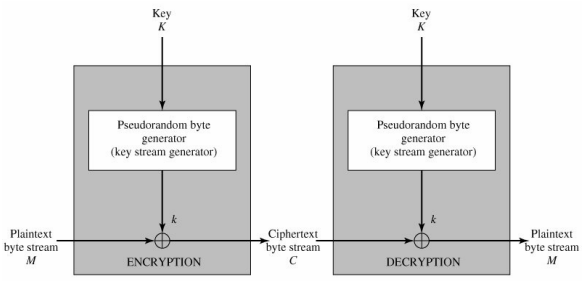
Skema algoritma protokol Diffie-Hellman pada Gambar 1. terdapat dua bilangan yang telah disepakati dan diketahui publik yaitu sebuah bilangan prima q dan sebuah bilangan α yang merupakan akar primitif dari q. Misalkan pengguna A dan B ingin bertukar kunci. Pengguna A memilih bilangan acak $X_A<q$ dan menghitung $Y_A = \alpha^{XA}$ mod q. Demikian pula, pengguna B secara bebas memilih bilangan bulat acak $X_B<q$ dan menghitung $Y_B = \alpha^{XB}$ mod q. Masing-masing pihak menyimpan nilai privat X_A dan X_B. Pengguna A memberikan nilai publik Y_A kepada pengguna B dan sebaliknya pengguna B memberikan nilai publik Y_B kepada pengguna A. Kemudian pengguna A menghitung kunci $K = (Y_B)^{XA}$ mod q dan pengguna B menghitung kunci $K = (Y_A)^{XB}$ mod q. Kedua perhitungan tersebut menghasilkan kunci yang sama.

Stallings juga menjelaskan bahwa protokol Diffie-Hellman memiliki kelemahan sewaktu pertukaran kunci sehingga terjadi suatu serangan yang disebut dengan *Man in the Middle Attack*. Serangan tersebut karena tidak adanya otentikasi dari para pengguna. Celah ini dapat diatasi dengan penggunaan tanda tangan digital dan sertifikat kunci publik.

V. ALGORITMA RC4

Schneier [15], menyatakan bahwa RC4 adalah suatu variable kunci *stream cipher* yang dikembangkan pada tahun 1987 oleh Ron Rivest untuk RSA Data Security, Inc. Selama tujuh tahun algoritma itu dirahasiakan karena adanya suatu perjanjian untuk menjaga kerahasiaanya. Pada bulan September 1994 seseorang telah mengirimkan *source code* RC4 ke milis *Cypherpunks*. Dengan cepat *source code* tersebut menyebar ke *newsgroup Usenet* yaitu *sci.crypt* dan Internet melalui situs-situs ftp di seluruh dunia. Kemudian RSA *Data Security*, Inc. mencoba menarik *source code* tersebut dengan mengklaim bahwa *source code* tersebut masih merupakan rahasia dagang mereka meskipun itu telah tersebar ke publik, tetapi itu sudah terlambat. Sejak itulah algoritma RC4 dibahas dan dibedah di *Usenet*, didistribusikan di konferensi dan diajarkan di kursus kriptografi.

Menurut Stallings [17], RC4 telah digunakan dalam SSL/TLS (*Secure Socket Layer*/*Transport Layer Security*) yang merupakan standar yang telah ditetapkan untuk komunikasi antara Web browser dan server. RC4 juga digunakan dalam protokol WEP (*Wired Equivalent Privacy*) dan yang lebih baru yaitu pada protokol *WiFi Protected Access* (WPA) yang merupakan bagian dari standar Wireless LAN IEEE 802.11. RC4 adalah algoritma yang sangat sederhana dan cukup mudah untuk menjelaskan. Sebuah variabel kunci yang panjangnya 1 sampai dengan 256 *byte* (8 sampai dengan 2.048 *bit*) digunakan untuk menginisialisasi 256 *byte* larik S, dengan elemen S[0], S[1], ..., S[255]. Setiap larik S berisi permutasi dari semua angka 8-*bit* mulai dari 0 sampai 255. Untuk enkripsi dan dekripsi, sebuah *byte* k (lihat Gambar 2.) dihasilkan dari larik S dengan memilih salah satu *byte* dari 255 *byte* secara sistematis. Karena setiap nilai k dihasilkan dari nilai larik S yang dipermutasi.



Gambar 2. Skema stream cipher RC4 (Stallings, 2005)

Untuk memulai inisialisasi larik S, maka ditetapkan nilai larik S sama dengan nilai dari 0 hingga 255 dalam urutan menaik yaitu; S[0]=0, S[1]=1, ..., S[255]=255. Kemudian diciptakan sebuah larik sementara yaitu larik T. Jika panjang kunci K adalah 256 *byte*, maka elemen K ditransfer ke elemen T. Jika tidak, tergantung dengan panjang kunci (*keylen*), maka elemen pertama dari T disalin dari elemen K dan kemudian diulangi sebanyak yang diperlukan untuk mengisi elemen T. Operasi tersebut dapat diringkas sebagai berikut:

```
for  i = 0 to 255 do
     S[i] = i;
     T[i] = K[i mod keylen];
```

Kemudian dilakukan permutasi larik S dengan menggunakan elemen larik T untuk menghasilkan permutasi awal larik S. Ini melibatkan mulai dari S[0] hingga S[255] dan untuk setiap larik S[i] menukar S[i] dengan *byte* lainnya menurut skema larik S yang ditentukan oleh T[i] seperti ringkasan dibawah ini:

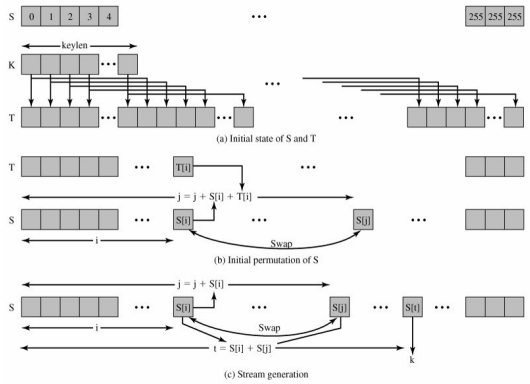
```
j = 0;
for  i = 0 to 255 do
     j = (j + S[i] + T[i]) mod 256;
     Swap (S[i], S[j]);
```

Karena operasi di atas adalah menukar letak atau permutasi maka nilai dari larik S masih tetap angka 0 sampai 255.

Setelah larik S diinisialisasi, maka kunci K tidak digunakan lagi. Kemudian dilakukan *stream generation* yang mencakup perputaran semua elemen S[i]. Untuk setiap elemen S[i], menukar S[i] dengan *byte* larik S lainnya berdasarkan skema yang telah ditentukan. Setelah S[255] tercapai, proses berlanjut dan dimulai lagi dari S[0]:

```
i, j = 0;
while (true)
    i = (i + 1) mod 256;
    j = (j + S[i]) mod 256;
    Swap (S[i], S[j]);
    t = (S[i] + S[j]) mod 256;
    k = S[t];
```

Untuk mengenkripsi, nilai k diXORkan dengan *byte* plainteks. Sedangkan untuk dekripsi, XORkan nilai k dengan *byte* cipherteks. Dari penjelasan algoritma di atas dapat diilustrasikan seperti yang terlihat pada Gambar 3.



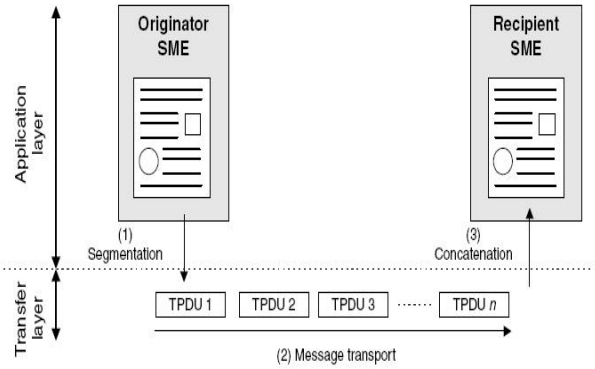
Gambar 3. Pseudorandom byte generator RC4 (Stallings, 2005)

VI. *SHORT MESSAGE SERVICE* (SMS)

Short Message Service (SMS) adalah layanan dasar yang memungkinkan pertukaran pesan teks singkat antara pelanggan. Pertama kali pesan teks singkat diyakini telah ditransfer pada tahun 1992 melalui saluran sinyal jaringan GSM Eropa. Karena ini sukses, maka penggunaan SMS telah menjadi subjek pertumbuhan yang luar biasa. SMS memungkinkan pengguna untuk bertukar pesan yang berisi teks singkat. Pesan-pesan ini dapat dikirim dari GSM/UMTS perangkat mobile tetapi juga dari berbagai perangkat lain seperti Internet, telex, dan faksimili. SMS adalah teknologi yang sangat matang didukung oleh 100% dari handset GSM dan oleh sebagian besar jaringan GSM di seluruh dunia [2].

VI.I STRUKTUR PESAN

Bodic [2], menerangkan bahwa sebuah pesan terdiri dari teks atau unsur-unsur seperti gambar, melodi dan lain-lain. Untuk keperluan transportasi dan karena keterbatasan pada lapisan transfer, maka aplikasi membagi pesan menjadi beberapa bagian yang disebut dengan segmen. Sebuah segmen pesan memiliki ukuran yang terbatas. Dalam rangka untuk menyampaikan sejumlah data, beberapa segmen pesan dapat dikombinasikan menjadi pesan yang digabungkan (*concatenated*). Penggabungan pesan tersebut ditangani di lapisan aplikasi. Supaya dapat diangkut, pesan segmen perlu dipetakan ke sebuah TPDU di lapisan transfer seperti Gambar 4.

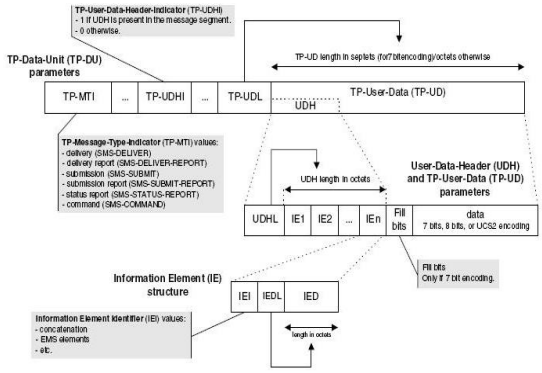


Gambar 4. Struktur pesan (Bodic, 2005)

Sebuah segmen pesan dikaitkan dengan sejumlah parameter. Parameter ini menunjukkan jenis pesan, kelas, kelompok coding, dan lain-lain. Selain itu, parameter juga mengandung isi pesan yang disediakan sendiri oleh pelanggan, penyedia konten, atau konten yang dihasilkan secara otomatis oleh sebuah mesin.

VI.II TRANSPORT PROTOCOL DATA UNIT (TPDU)

TPDU melaksanakan setiap transaksi yang dilakukan antara SME dan SMSC pada lapisan transfer. Struktur dari TPDU dapat dilihat seperti pada Gambar 5 [2].



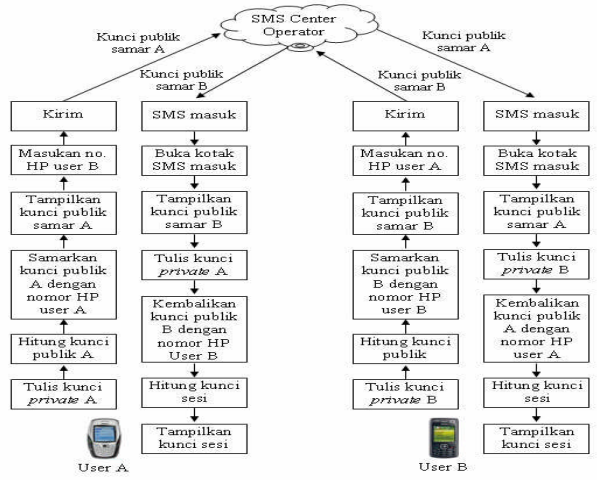
Gambar 5. Struktur TPDU (Bodic, 2005)

VI.III SKEMA KODE TEKS

Bodic [2], menerangkan bahwa teks merupakan bagian dari pesan yang bisa dikodekan menurut beberapa abjad teks. Dua skema pengkodean teks yang dapat digunakan dalam SMS adalah GSM 7-*bit* alfabet standar dan *Universal Character Set* (UCS2). Sedangkan jumlah teks 140 karakter yang terdapat dalam segmen pesan merupakan skema kode 8-*bit*. *Universal Character Set* (UCS2) digunakan untuk pengkodean kompleks set karakter non-Latin (16-*bit*) seperti Cina dan Arab.

VII.RANCANGAN SISTEM

Rancangan sistem adalah suatu gambaran dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah yang disatukan ke dalam satu kesatuan yang utuh. Dalam sistem yang akan dibangun, secara umum memiliki dua buah rancangan yaitu rancangan pertukaran kunci (Gambar 6.) dan rancangan enkripsi dan dekripsi pesan (Gambar 8.).



Gambar 6. Rancangan pertukaran kunci

Gambar 6. merupakan rancangan pertukaran kunci berbasis SMS. Algoritma pertukaran kunci yang digunakan dalam sistem ini adalah algoritma pertukaran kunci Diffie-Hellman, dimana terdapat modifikasi pada perhitungan kunci publik dan kunci sesi dari algoritma tersebut (Gambar 7.). Tujuan modifikasi adalah untuk mengatasi komputasi yang besar dan tidak adanya otentikasi dari perhitungan protokol Diffie-Hellman yang asli.

Gambar 6. dapat dijelaskan bahwa user A menulis kunci *private* (X_A) dimana nilainya ditentukan oleh user A dan memiliki batasan mulai dari 0 sampai 999999999. Kemudian sistem melakukan perhitungan kunci publik A (Y_A), tetapi sebelumnya terlebih dahulu kunci *private* (X_A) disamarkan ($Z_A = X_A \bmod p$) oleh sistem tujuannya supaya nilai kunci *private* (X_A) yang besar dapat diperkecil. Pada proses penyamaran nilai kunci *private* A (Z_A) terdapat variabel p yang nilainya adalah bilangan asli dan harus kecil dari nilai q. Pada Gambar 7.(b) dapat dilihat bahwa terdapat juga variabel α dan q dimana nilainya adalah bilangan prima, nilai α harus kecil dari nilai q. Setelah dilakukan beberapa kali percobaan pada pemrograman maka didapat nilai q maksimal 19, nilai p maksimal 16 dan nilai α maksimal harus kecil dari nilai q. Nilai α, p dan q ditentukan atau diset pada sistem.

Setelah kunci *private* A disamarkan nilainya (Z_A) oleh sistem maka dilakukan perhitungan kunci publik A ($Y_A = \alpha^{Z_A} \bmod q$). Kemudian kunci publik A disamarkan (Y_A^1) oleh sistem dengan cara mengkombinasikan dengan nomor HP user A ($Y_A^1 = Y_A * HP_A * V_A$) dimana nilai nomor HP user A diambil dari setting nomor HP yang telah disimpan pada HP, setelah itu kunci publik samar A (Y_A^1) dibalik posisinya ($Y_A^1 = \text{Balik}(Y_A^1)$) oleh sistem. HP_A digunakan untuk otentikasi kunci publik dimana nilainya diambil 9 digit sebelah kanan dari nomor HP user A dengan alasan nomor telepon seluler GSM di Indonesia yang terpendek adalah 10 digit ketika nomor telepon seluler tersebut diawali dengan angka 0 dan 12 digit ketika diawali dengan +62 (kode negara Indonesia). Jika diambil 10 digit sebelah kanan maka digit pertama pemenggalan akan terjadi perbedaan yaitu angka 0 atau angka 2. Sedangkan variabel V_A tujuannya untuk lebih menyamarkan nilai kunci publik A (Y_A) dimana nilainya diambil 2 digit sebelah kiri dari nilai HP_A. Alasan diambil 2 digit sebelah kiri dari nilai HP_A adalah untuk menghindari kemungkinan nilai V_A menjadi nol. Karena nilai HP_A tidak

diawali oleh angka nol, tetapi jika diambil diposisi tengah atau posisi kanan dari nilai HP_A ada kemungkinan nilai V_A menjadi nol sehingga akan menghasilkan nilai nol pada perhitungan kunci publik samar A (Y_A^1). Sebagai contoh nilai HP_A = 324500500 jika diambil 2 digit sebelah kanan maka akan didapat nilai stringnya 00 dan jika dikonversi ke integer maka nilainya menjadi 0.

Setelah dilakukan penyamaran kunci publik A (Y_A^1) oleh sistem maka sistem akan menampilkan hasil kunci publik samar A (Y_A^1). Kemudian user A memasukan nomor HP user B sebagai nomor tujuan pengiriman kunci publik samar A. Setelah itu sistem mengirim kunci publik samar A (Y_A^1) ke nomor HP user B melalui SMS Center Operator.

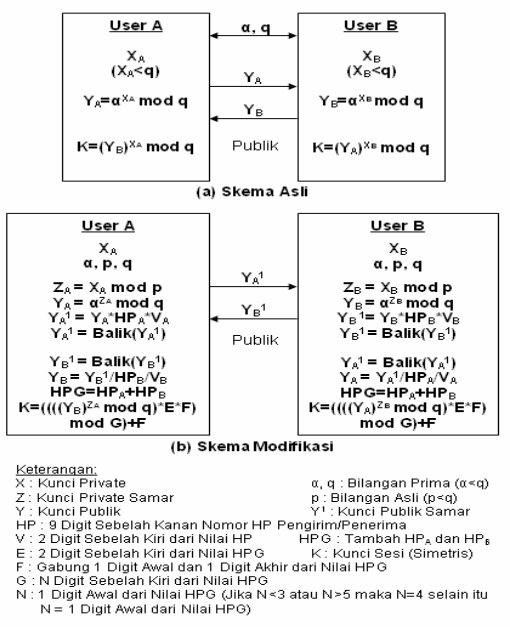
Kemudian sistem user B menerima SMS masuk berupa kunci publik samar A (Y_A^1). Setelah itu user B membuka kotak SMS masuk dan sistem menampilkan kunci publik samar A (Y_A^1). Kemudian user B memasukan atau menulis kunci *private* B (X_B) dimana batasan nilainya adalah mulai dari 0 sampai 999999999. Kemudian sistem melakukan penyamaran kunci *private* B ($Z_B = X_B \bmod p$), kunci *private* samar B (Z_B) nantinya akan digunakan untuk perhitungan kunci sesi K oleh sistem, dimana nilai p dan q yang digunakan pada perhitungan kunci *private* B (Z_B) dan perhitungan kunci sesi K sama nilainya dengan nilai p dan q waktu menyamarkan kunci *private* A (Z_A) dan perhitungan kunci publik A (Y_A).

Sebelum kunci sesi K dihitung maka kunci publik samar A (Y_A^1) dikembalikan dulu menjadi kunci publik A (Y_A) oleh sistem dengan cara membalikan posisi kunci publik samar A (Y_A^1 = Balik(Y_A^1). Kemudian sistem mengkombinasikan kembali kunci publik samar A (Y_A^1) yang telah dibalikan posisinya dengan nomor HP user A ($Y_A = Y_A^1/HP_A/V_A$), dimana nomor HP user A nilainya diambil dari nomor HP pengirim sewaktu SMS masuk. Sedangkan nilai HP_A diambil 9 digit dari nomor HP user A dan nilai V_A diambil 2 digit dari nilai HP_A.

Setelah kunci publik A (Y_A) yang asli telah dikembalikan maka dilakukan penggabungan nomor HP ($HPG=HP_A+HP_B$) oleh sistem tujuannya supaya nilai penggabungan HP (HPG) dapat digunakan untuk perhitungan kunci sesi sekaligus sebagai otentikasi dari perhitungan kunci sesi K. Setelah didapat nilai dari penggabungan HP (HPG) maka sistem melakukan perhitungan kunci sesi K ($K=((((Y_A)^{Z_B} \bmod q)*E*F) \bmod G)+F$). Pada perhitungan kunci sesi K terdapat variabel E dimana nilainya diambil 2 digit sebelah kiri dari nilai HPG tujuannya supaya tidak menghasilkan nilai nol sama halnya waktu menentukan nilai V_A. Kemudian terdapat juga variabel F dimana nilainya adalah gabungan antara 1 digit awal dan 1 digit akhir dari nilai HPG yang tujuannya untuk membedakan dengan nilai variabel E dan menghindari nilai nol. Variabel E dan F digunakan untuk memperbesar hasil perhitungan kunci sesi asli ($(Y_A)^{Z_B} \bmod q$). Kemudian juga terdapat variabel G dimana nilainya diambil sebanyak N digit dari nilai HPG. Variabel N nilainya diambil 1 digit awal dari nilai HPG, jika nilai N<3 atau N>5 maka nilai N diset menjadi 4 pada sistem selain itu nilai N sama nilainya dengan 1 digit awal dari nilai HPG. Tujuan dari adanya variabel G adalah untuk membatasi hasil perkalian ($(Y_A)^{Z_B} \bmod q$)*E*F supaya tidak terlalu besar yaitu dengan cara (($(Y_A)^{Z_B} \bmod q$)*E*F) mod G. Karena kunci sesi K yang akan digunakan pada proses enkripsi dan dekripsi mempunyai nilai maksimal 999999. Kemudian terdapat penambahan variabel F pada akhir perhitungan kunci sesi K, tujuannya untuk menjaga supaya hasil perhitungan kunci sesi K tidak menghasilkan nilai nol. Karena sewaktu melakukan perhitungan (($(Y_A)^{Z_B} \bmod q$)*E*F) mod G ada kemungkinan menghasilkan nilai nol. Setelah dilakukan perhitungan kunci sesi K maka sistem akan menampilkan kunci sesi hasil pertukaran dengan user A yang akan digunakan oleh user B untuk proses enkripsi dan dekripsi pesan SMS berikutnya.

Setelah user B mendapatkan kunci sesi K maka user B melakukan hal yang sama seperti yang dilakukan user A sehingga nantinya setelah pengiriman kunci publik samar B (Y_B^1) dan sistem melakukan perhitungan kunci sesi K maka user A akan mendapatkan nilai kunci sesi K yang sama dengan user B seperti yang digambarkan pada Gambar 7.(b).

Pada sistem yang akan dibangun harus terdapat suatu port untuk komunikasi (pengiriman dan penerimaan) SMS. Dimana nilai port tersebut harus sama waktu pengiriman dan penerimaan SMS. Port tersebut akan *include* dalam nomor HP sewaktu pengiriman SMS. Nilai port tersebut bisa sembarangan yang penting nilainya berupa angka dan harus diset pada sistem. Jika tidak ada nomor port tersebut maka pesan SMS yang dikirm akan diterima oleh aplikasi SMS standar dari HP penerima.



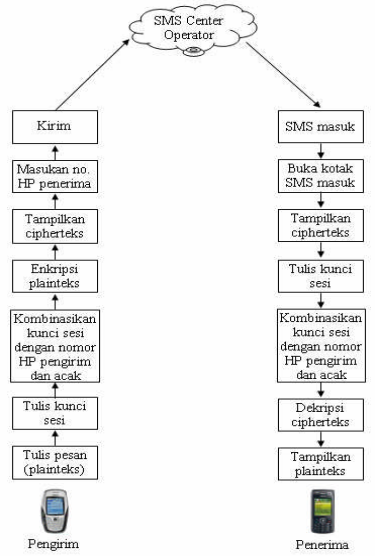
Gambar 7. Rancangan modifikasi protokol Diffie-Hellman

Gambar 7. dapat dilihat bahwa terdapat modifikasi pada protokol Diffie-Hellman yang asli. Dimana pada skema asli terdapat nilai α dan q yang nilainya disepakati oleh kedua user dan diketahui oleh publik. Sedangkan skema modifikasi terdapat nilai α, p dan q dimana nilainya diset pada sistem dan tidak diketahui oleh publik maupun kedua user. Kemudian pada skema asli nilai kunci *private* X kecil dari nilai q, sedangkan pada skema modifikasi kunci *private* X nilainya dari 0 sampai dengan 999999999 yang kemudian nilainya disamarkan dengan cara Z = X mod p.

Pada skema asli kunci publik Y setelah dihitung langsung dikirim kepada user penerima, sedangkan pada skema modifikasi setelah kunci publik Y dihitung kemudian disamarkan terlebih dahulu dengan nomor HP user pengirim ((Y^1 = Y*HP*V) dan (Y^1 = Balik(Y^1)) kemudian kunci publik samar (Y^1) baru dikirim kepada user penerima. Setelah itu pada skema asli kunci publik (Y) dari user pengirim diterima maka langsung dihitung kunci sesi K, sedangkan pada skema modifikasi kunci publik samar (Y^1) yang diterima harus dikembalikan dulu menjadi kunci publik asli (Y) dengan mengkombinasikan lagi dengan nomor HP user pengirim ((Y^1 = Balik(Y^1) dan (Y = Y^1/HP/V)) kemudian baru dihitung kunci sesi K seperti yang telah dijelaskan pada rancangan pertukaran kunci.

Pada skema modifikasi protokol Diffie-Hellman (Gambar 7.(b)) terdapat perhitungan kunci sesi K oleh user A yaitu K=(((Y_B)^{Z_A} mod q)*E*F) mod G)+F dan user B yaitu K=(((Y_A)^{Z_B} mod q)*E*F) mod G)+F dimana kedua perhitungan tersebut akan mempunyai hasil yang sama dan dapat dibuktikan secara matematis sebagai berikut:

$$
\begin{aligned}
K &= ((((Y_B)^{Z_A} \bmod q)*E*F) \bmod G)+F \\
&= ((((\alpha^{Z_B} \bmod q)^{Z_A} \bmod q)*E*F) \bmod G)+F \\
&= ((((\alpha^{Z_B})^{Z_A} \bmod q)*E*F) \bmod G)+F \\
&= (((\alpha^{Z_B Z_A} \bmod q)*E*F) \bmod G)+F \\
&= (((\alpha^{Z_A Z_B} \bmod q)*E*F) \bmod G)+F \\
&= ((((\alpha^{Z_A})^{Z_B} \bmod q)*E*F) \bmod G)+F \\
&= ((((\alpha^{Z_A} \bmod q)^{Z_B} \bmod q)*E*F) \bmod G)+F \\
&= ((((Y_A)^{Z_B} \bmod q)*E*F) \bmod G)+F
\end{aligned}
$$



Gambar 8. Rancangan enkripsi dan dekripsi pesan

Gambar 8. merupakan rancangan enkripsi dan dekripsi pesan SMS. Awalnya pengirim menulis pesan kemudian pengirim menulis kunci sesi K, dimana nilai kunci sesi K berasal dari hasil pertukaran kunci menggunakan modifikasi algoritma Diffie-Hellman yang telah dijelaskan pada rancangan pertukaran kunci. Setelah pengirim menulis kunci sesi K maka sistem melakukan kombinasi antara kunci sesi K dengan nomor HP pengirim yang telah disetting dan telah disimpan pada HP (K1=K*HP dan K2=K*HP*V) (Gambar 9.(b)) tujuannya sebagai otentikasi dan memperpajang ukuran kunci. Variabel HP nilainya diambil 9 digit sebelah kanan dari nilai HP pengirim dan variabel V nilainya diambil 2 digit sebelah kiri dari nilai HP tujuannya sama dengan waktu melakukan

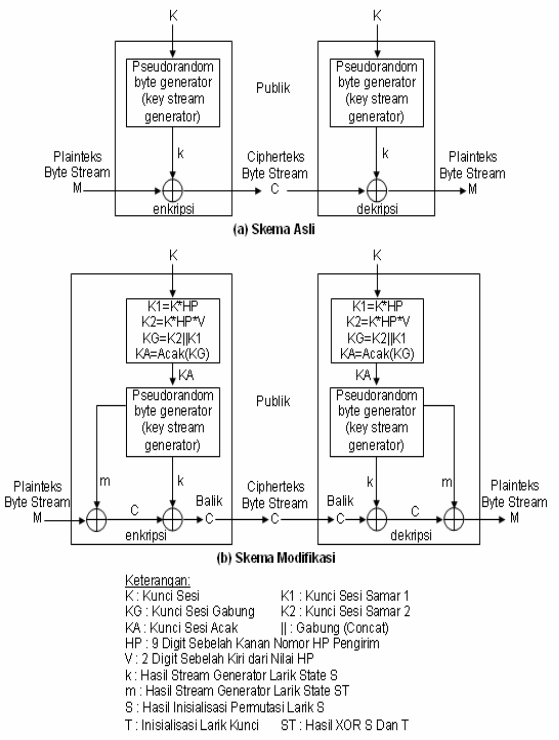
perhitungan pertukaran kunci. Kemudian sistem menggabung K2 dengan K1 (KG=K2||K1) dan dilakukan pengacakan (KA=Acak(KG)). Pengacakan kunci digunakan untuk lebih memperpanjang ukuran kunci yang telah digabung sebelumnya sehingga mendekati atau mencapai 256 *byte*.

Setelah pengacakan kunci sesi maka sistem melakukan proses enkripsi dan menampilkan hasil enkripsi (cipherteks) kemudian pengirim memasukan nomor HP tujuan dan sistem mengirim cipherteks melalui SMS Center Operator. Kemudian pada HP penerima SMS masuk dan penerima membuka kotak SMS masuk dan sistem menampilkan pesan cipherteks. Setelah itu penerima memasukan kunci sesi K yang sama dengan kunci sesi K pengirim.

Setelah penerima menulis kunci sesi K maka sistem melakukan kombinasi antara kunci sesi K dengan nomor HP pengirim yang ada pada SMS masuk (K1=K*HP dan K2=K*HP*V) (Gambar 9.(b)) tujuannya sebagai otentikasi dan memperpajang ukuran kunci. Variabel HP nilainya diambil 9 digit sebelah kanan dari nilai HP pengirim dan variabel V nilainya diambil 2 digit sebelah kiri dari nilai HP tujuannya sama dengan waktu melakukan perhitungan pertukaran kunci. Kemudian sistem menggabung K2 dengan K1 (KG=K2||K1) dan dilakukan pengacakan kunci (KA=Acak(KG)). Setelah pengacakan kunci maka sistem melakukan proses dekripsi dan menampilkan hasil dekripsi (plainteks).

Algoritma yang digunakan untuk proses enkripsi dan dekripsi adalah modifikasi algoritma RC4. Modifikasi dilakukan pada kunci sesi (Gambar 9.(b)) seperti yang telah dijelaskan di atas. Setelah dilakukan kombinasi dan pengacakan kunci maka sistem melakukan proses *pseudorandom byte generator* (Gambar 10.).

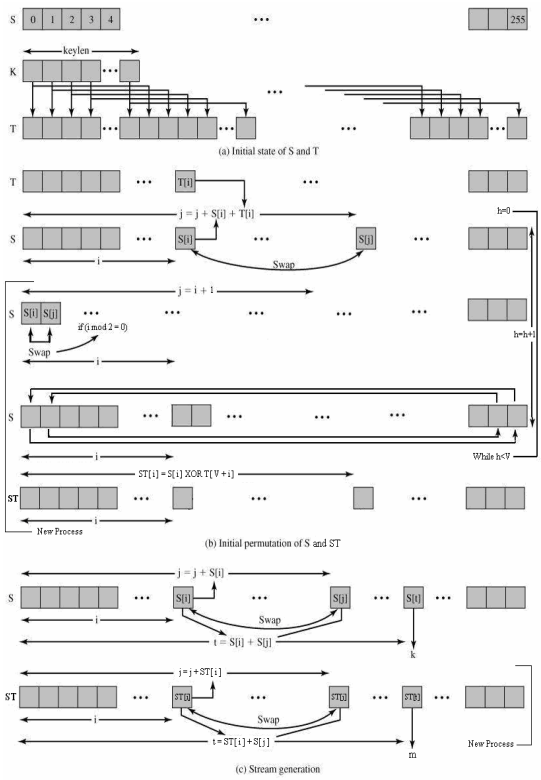
Pada proses *pseudorandom byte generator* tepatnya pada inisialisasi permutasi S dan ST (Gambar 10.(b)) terdapat tiga buah proses baru yaitu proses pertukaran posisi genap ke ganjil dan sebaliknya dari larik S, proses membalik posisi larik S dan proses inisialisasi larik ST dengan cara mengXORkan nilai larik S dan T. Pada proses *pseudorandom byte generator* juga terdapat penambahan proses pada stream generation untuk larik ST (Gambar 10.(c)).



Gambar 9. Rancangan modifikasi stream cipher RC4

Proses permutasi larik S, pertukaran posisi genap ke ganjil dan sebaliknya dari larik S serta proses membalik posisi larik S akan diulang sebanyak nilai V. Tujuannya untuk lebih mengacak posisi larik S sehingga akan sulit dilacak posisi dari nilai larik S. Setelah proses tersebut, larik S masih memiliki nilai tetap dari angka 0 sampai 255 tetapi posisinya makin acak. Sedangkan proses inisialisasi larik ST digunakan untuk menyamarkan kunci sesi acak (larik T) dimana nanti nilai dari larik ST akan digunakan untuk proses enkripsi dan dekripsi. Kemudian penambahan proses stream generation larik ST bertujuan untuk mengacak larik ST pada waktu proses enkripsi dan dekripsi, proses tersebut hampir sama tujuannya dengan proses stream generation pada larik S.

Setelah proses *pseudorandom byte generator* maka dilakukan proses enkripsi atau proses dekripsi. Pada algoritma RC4 yang asli (Gambar 9.(a)) proses enkripsi dan dekripsi hampir sama yang membedakan hanya aliran karakter plainteks dan cipherteks yang akan diXOR dengan nilai k. Sedangkan pada modifikasi RC4 proses enkripsi dan dekripsi berbeda karena terdapat dua proses XOR setelah proses *pseudorandom byte generator* dan pembalikan posisi karakter.



Gambar 10. Modifikasi Pseudorandom byte generator RC4

Pada proses enkripsi RC4 yang telah dimodifikasi *byte stream* plainteks akan diXORkan terlebih dahulu dengan *byte* m (hasil stream generation larik ST) dan terbentuk *byte stream* cipherteks. Kemudian *byte stream* cipherteks tersebut diXORkan lagi dengan *byte* k (hasil stream generation larik S), setelah itu terbentuk *byte stream* cipherteks yang baru. Setelah itu *byte stream* cipherteks dibalik posisi karakternya.

Pada proses dekripsi RC4 yang telah dimodifikasi *byte stream* cipherteks dibalik lagi posisi karakternya sehingga kembali ke posisi semula. Kemudian *byte stream* cipherteks tersebut diXORkan dengan *byte* k dan terbentuk *byte stream* cipherteks baru. Setelah itu *byte stream* cipherteks yang baru diXORkan lagi dengan *byte* m sehingga akan terbentuk *byte stream* plainteks (teks asli).

Tujuan dari pengacakan kunci yang berlapis, proses XOR yang berlapis dan proses pemutaran posisi teks pada enkripsi dan dekripsi adalah untuk menghindari serangan *know-plaintext attack* dan *ciphertext-only attack*, karena algoritma *stream cipher* rentan dari kedua serangan tersebut terutama pada algoritma RC4 yang asli rentan dengan serangan *know-plaintext attack*. Serangan *know-plaintext attack* bisa diartikan jika kriptanalisis memiliki potongan plainteks dan cipherteks maka ia akan dapat menemukan aliran kunci dengan cara mengXORkan plainteks dan cipherteks. Serangan *ciphertext-only attack* bisa diartikan jika kriptanalisis memiliki dua potong cipherteks maka ia akan dapat menemukan aliran kunci dengan cara mengXORkan kedua cipherteks tersebut kemudian menghasilkan dua buah plainteks, setelah itu plainteks dikombinasikan dengan cipherteks dan setelah itu kriptanalisis bisa menerka kunci yang digunakan, tetapi serangan ini bisa terjadi jika menggunakan kunci yang sama. Jadi dengan terdapatnya modifikasi pada algoritma RC4, seorang kriptanalis akan susah untuk membongkar pesan tanpa kunci.

VIII. IMPLEMENTASI

Implementasi dari semua proses dalam perancangan sistem dinyatakan dengan menggunakan bahasa pemrograman Java dalam platform Micro Edition (J2ME) yang berbasis MIDP 2.0 dan CLDC 1.0. Dalam pemrograman J2ME menggunakan perangkat lunak NetBeans IDE 6.0. sekaligus sebagai *emulator*. Sedangkan untuk implementasi pada telepon seluler digunakan telepon seluler berbasis Java yang mendukung MIDP 2.0 dan CLDC 1.0.

Sebelum melakukan implementasi program pada telepon seluler ada beberapa hal yang harus diperhatikan sewaktu melakukan pemrograman yaitu pengaturan konfigurasi *platform* program dan port SMS pada perangkat lunak NetBeans IDE 6.0. Pada konfigurasi *platform* program yang perlu diatur adalah *Device Configuration* dengan konfigurasi CLDC-1.0 dan *Device Profile* dengan konfigurasi MIDP-2.0. Sedangkan untuk port SMS dikonfigurasi dengan nilai 50000.

IX. SIMULASI PENGUJIAN PERTUKARAN KUNCI DISERANG PADA SMS

Protokol Diffie-Hellman yang asli memiliki kelemahan yaitu tidak adanya otentikasi dari pertukaran kunci. Sehingga rentan terhadap serangan terutama *Man in the Middle Attack* yaitu suatu serangan aktif dimana penyerangnya menyadap dan merubah pesan dari pengirim ke penerima pesan, seolah-olah pesan tersebut berasal dari pengirim aslinya.

Setelah adanya modifikasi protokol Diffie Hellman jika terdapat pembajakan dan perubahan kunci publik oleh user lain yang memakai nomor telepon seluler selain nomor telepon seluler user A dan B maka akan langsung ketahuan bahwa nomor pengirim berbeda sewaktu pesan SMS (kunci publik) masuk. Jadi tidak akan mungkin penyerang bisa merubah kunci publik dengan menggunakan sistem yang dibangun dan nomor telepon yang berbeda dari kedua user A dan B tersebut. Kecuali ada kemungkinan pesan (kunci publik) dicegat dan dirubah pada SMS Center dari operator seluler, tetapi perubahan itupun hanya mengakibatkan kesalahan pada otentikasi dari kunci publik sehingga perhitungan kunci sesi tidak bisa dilakukan karena kunci publik dianggap sistem tidak dapat dipercaya. Tabel 1. merupakan tabel simulasi pengujian pertukaran kunci yang diserang dengan *Man in the Middle Attack* pada SMS Center.

Tabel 1. Simulasi pengujian pertukaran kunci diserang pada SMS Center

Kunci publik A yang seharusnya diterima B	Kunci publik A yang telah diserang dan diterima B	Kunci publik B yang seharusnya diterima A	Kunci publik B yang telah diserang dan diterima A	Hasil pertukaran (kunci sesi bisa dihitung/tidak)	
				A	B
05834295357	73293472938	27734541153	22381232311	Tidak	Tidak
07784817051	5283920123	44578092207	44578092203	Tidak	Tidak
01364551254	01364551251	65379997155	65379997154	Tidak	Tidak
04439138308	48920283831	29762723001	39762723042	Tidak	Tidak
05974791152	25974791156	6933636105	43216369872	Tidak	Tidak

Pada Tabel 1. dapat dilihat bahwa jika terjadi serangan dengan cara merubah kunci publik maka kunci sesi tidak dapat dihitung. Jika salah satu kunci publik diserang atau dirubah maka hanya satu user yang mendapatkan kunci sesi tetapi nilainya akan salah dan user yang lain tidak dapat menghitung kunci sesi, sehingga mengakibatkan komunikasi menjadi satu arah.

Kemudian dalam proses pertukaran kunci terdapat perhitungan dan penyamaran kunci publik serta nomor telepon seluler sebagai otentikasi sehingga dapat menjaga kerahasiaan, otentikasi, integritas dan penyangkalan dari kunci yang ditukarkan.

X. SIMULASI PENGUJIAN SERANGAN KNOW-PLAINTEXT ATTACK

Algoritma RC4 yang asli rentan dengan serangan *know-plaintext attack*. Serangan *know-plaintext attack* bisa diartikan jika kriptanalisis memiliki potongan plainteks dan cipherteks maka ia akan dapat menemukan aliran kunci dengan cara mengXORkan plainteks dan cipherteks dengan syarat kunci untuk menghasilkan cipherteks yang digunakan adalah sama. Kemudian kunci tersebut digunakan untuk membuka cipherteks yang lain dengan cara mengXORkan kunci dengan cipherteks tersebut. Berikut ini adalah simulasi serangan *know-plaintext attack*, dimana kunci sesi yang digunakan adalah sama. Sedangkan nomor telepon seluler yang digunakan adalah +6285228016518.

Plainteks

Iya pak, saya berangkat ke malang ntar malam. Makasih atas bantuannya

ASCII dalam desimal:

73	121	97	32	112	97	107	44	32	115	97	121
97	32	98	101	114	97	110	103	107	97	116	32
107	101	32	109	97	108	97	110	103	32	110	116
97	114	32	109	97	108	97	109	46	32	77	97
107	97	115	105	104	32	97	116	97	115	32	98
97	110	116	117	97	110	110	121	97			

Plainteks diketahui oleh penyerang

Cipherteks

-ýÎ7 -6JÎÐH5ž)%)"¾ 1k‰dÕ®ª~˜Ìp□Îı²Éž!
äbÎN}R³ <ZŽA^¤

ASCII dalam desimal:

127	173	253	206	55	150	54	74	206	208	72	16
53	158	41	37	41	148	190	145	107	137	100	16
213	174	176	24	126	3	152	204	112	63	206	236
178	201	158	33	228	98	25	206	78	125	82	170
139	90	142	22	65	152	164	2				

Cipherteks diketahui oleh penyerang

Cipherteks diatas hasil enkripsi dari kalimat: Besok tolong kamu transfer uang sebesar Rp. 10.000.000,- (plainteks tidak diketahui penyerang)

Kunci (plainteks XOR cipherteks)

ASCII dalam desimal:

54	212	156	238	71	247	93	102	238	163	41	105
84	190	75	64	91	245	208	246	0	232	16	48
190	203	144	117	31	111	249	162	23	31	160	152
211	187	190	76	133	14	120	163	96	93	31	203
224	59	253	127	41	184	197	118				

Cipherteks1

=ő¢ŒeˆmŽÞLtÞyh'¾Â"Ó6UÀýìH:Ú‰8Œ
Ő åàÈÓ5â#^□HX¬ÄFÌ€µ

ASCII dalam desimal:

61	245	162	140	101	152	109	27	142	222	25	76
116	222	121	4	104	146	190	194	34	211	54	85
192	253	236	72	58	27	218	137	56	140	213	229
224	200	211	53	226	35	94	63	72	127	88	172
196	70	204	17	14	128	181	21				

Cipherteks diketahui oleh penyerang

Cipherteks diatas hasil enkripsi dari kalimat: Utk lbh jelas, kamu hub. 0811245678, atas nama pak broto (plainteks tidak diketahui penyerang)

Plainteks (kunci XOR cipherteks1)

ASCII dalam desimal:

11	33	62	98	34	111	48	125	96	125	48	37
32	96	50	68	51	103	110	52	34	59	38	101
126	54	124	61	37	116	35	43	47	147	117	125
51	115	109	121	103	45	38	156	40	34	71	103
36	125	49	110	39	56	112	99				

Plainteks hasil serangan *know-plaintext attack*

Plainteks asli dari cipherteks1
Utk lbh jelas, kamu hub. 0811245678, atas nama pak broto
ASCII dalam desimal:

85	116	107	32	108	98	104	32	106	101	108	97
115	44	32	107	97	109	117	32	104	117	98	46
32	48	56	49	49	50	52	53	54	55	56	44
32	97	116	97	115	32	110	97	109	97	32	112
97	107	32	98	114	111	116	111				

Jadi dapat dilihat bahwa plainteks hasil serangan berbeda dengan plainteks asli dari cipherteks1. Dengan adanya modifikasi pada algoritma RC4 maka terbukti dapat mencegah sera ngan *know-plaintext attack*. Kemudian dengan adanya nomor telepon seluler yang digunakan sebagai otentikasi kunci, maka dapat menjaga kerahasiaan, integritas, otentikasi dan penyangkalan pada pesan yang dikirim.

XI. PENGUJIAN PANJANG PESAN SMS

Pada pertukaran kunci, pengiriman pesan hanya dalam 1 segmen pesan, karena kunci publik yang dikirim hanya terdiri dari beberapa karakter yaitu antara 10 sampai 12 karakter (*byte*). Karakter kunci publik tersebut berupa angka yang dapat dikenal oleh alfabet 7 *bit* standar GSM. Untuk itu pengujian panjang pesan SMS hanya akan dilakukan pada hasil proses enkripsi yang akan dikirim.

Pada pengujian panjang SMS jika terdapat satu atau beberapa karakter yang tidak dikenali oleh alfabet 7 *bit* standar GSM seperti karakter ASCII dalam desimal 237, 150 dan 154 maka semua kumpulan karakter hasil enkripsi tersebut akan dirubah menjadi karakter 16 *bit* (UCS2) walaupun terdapat karakter yang dikenal oleh alfabet 7 bit standar GSM dalam hasil enkripsi tersebut seperti karakter ASCII dalam desimal 64, 117 dan 43. Sehingga jumlah *bit* yang dihasilkan akan menjadi dua kali lipat dari jumlah *bit* pesan asli. Oleh karena itu menyebabkan terjadinya pemborosan dalam pengiriman pesan.

XII. KESIMPULAN

Berdasarkan hasil pembangunan sistem ini dan pembahasan pada bab-bab yang tersaji sebelumnya pada laporan ini, maka dapat ditarik kesimpulan yaitu:

1. Dengan adanya modifikasi protokol Diffie-Hellman yang menyertakan otentikasi maka proses pertukaran kunci dapat diamankan dari serangan *Man in the Middle Attack*.

2. Dengan adanya modifikasi algoritma RC4 maka pesan dapat diamankan dari serangan *know-plaintext attack*.

3. Dengan adanya nomor telepon seluler sebagai otentikasi, kunci, proses enkripsi dan dekripsi maka secara otomatis dapat menjaga *confidentiality* (kerahasiaan), *data integrity* (integritas data), *authentication* (otentikasi) dan *non-repudiation* (penyangkalan) pada pesan.

4. Sistem tidak akan berjalan jika salah satu user tidak mengaktifkan sistem, karena pesan (kunci publik atau cipherteks) yang dikirim akan masuk ke aplikasi SMS standar dari telepon seluler.

5. Jika hasil enkripsi pesan (cipherteks) terdapat karakter-karakter yang tidak dikenali oleh alfabet 7 *bit* standar GSM waktu pengiriman pesan maka semua karakter ciperteks tersebut dirubah menjadi karakter 16 *bit* (UCS2). Oleh karena itu terjadi pemborosan dalam pengiriman pesan SMS yang mengakibatkan pemakaian pulsa juga jadi boros.

XIII.SARAN

Adapun saran yang dapat dimanfaatkan bagi pengembangan perangkat lunak ini adalah menambahkan sistem kompresi dalam sistem yang dibangun, sehingga pemborosan pulsa dapat diminimalisir.

XIV. DAFTAR PUSTAKA

[1] Bishop, M., 2004, *Introduction to Computer Security*, Prentice Hall PTR, New Jersey.

[2] Bodic, G., L., 2005, *Mobile Messaging Technologies and Services SMS, EMS and MMS Second Edition*, John Wiley & Sons Ltd. England.

[3] Carts, D.A., 2001, A Review of the Diffie-Hellman Algorithm and its Use in Secure Internet Protocols, *SANS Institute Infosec Reading Room*, The SANS Institute, Maryland, USA.

[4] Chandra, P., 2005, *Bulletproof Wireless Security - GSM, UMTS, 802.11 and Ad Hoc Security*, Newnes Elsevier Inc., USA.

[5] Enany, A., 2007, Achieving Security in Messaging and Personal Content in Symbian Phone, *Thesis*, Department of Interaction and System Design School of Engineering, Blekinge Institute of Technology, Sweden.

[6] Fahmy, A., 2005, Key Exchange Protocol over Insecure Channel, *Proceedings of World Academy of Science, Engineering and Technology*, Volume 6:2005, Juni 2005, pp.34-36, Oslo, Norway.

[7] Hamilton, K. and Miles, R., 2006, *Learning UML 2.0*, O'Reilly, England.

[8] Hermawan, A. P., 2009, Kompresi Pesan SMS Menggunakan Algoritma Modified Half byte, *Tesis*, Ilmu Komputer, Universitas Gadjah Mada, Yogyakarta.

[9] Khine, L., L., 2009, A New Variant of RC4 Stream Cipher, *Proceedings of World Academy of Science, Engineering and Technology*, Volume 50:2009, Februari 2009, pp.958-961, Penang, Malaysia.

[10] Mahalanobis, A., 2005, Diffie-Hellman Key Exchange Protocol, Its Generalization and Nilpotent Groups, *Disertasi*, Faculty of The Charles E. Schmidt College of Science, Florida Atlantic University, Boca Raton, Florida.

[11] Menezes, A., Van Oorschot, P. and Vanstone, S., 1997, *Handbook of Applied Cryptography*, CRC Press, USA.

[12] Mousa, A. dan Hamad, A., 2006, Evaluation of the RC4 Algorithm for Data Encryption, *International Journal of Computer Science & Applications*, No.2, Vol.3, June 2006, pp. 44-56, An-Najah University, Nablus, Palestine.

[13] Oppliger, R., 2005, *Contemporary Cryptography*, Artech House, Inc., Boston, London.

[14] Ratshinanga, H., Lo, J dan Bishop, J., 2004, A Security Mechanism for Secure SMS Communication, *Proceedings of SAICSIT*, South Africa.

[15] Schneier, B., 1996, *Applied Cryptography – Protocol, Algorithms, and Source Code in C Second Edition*, John Wiley & Sons, Inc., United Kingdom.

[16] Setianto, Y. B. D., 2008, Distribusi Kunci Berbasis Short Message Service, *Tesis*, Ilmu Komputer, Universitas Gadjah Mada, Yogyakarta.

[17] Stallings, W., 2005, *Cryptography and Network Security Principles and Practices Fourth Edition*, Prentice Hall, New Jersey.

[18] Ye, N., 2008, *Secure Computer and Network - Systems Modeling, Analysis and Design*, John Wiley & Sons Ltd., England.