Pembangunan *Add-on* pada *Mozilla Thunderbird* untuk Enkripsi Surat Elektronik dengan *Corrected Block Tiny Encryption Algorithm*

Ricky Gilbert Fernando – 13505077
Program Studi Teknik Informatika, Institut Teknologi Bandung, Jl. Ganesha 10, Bandung
E-mail : if15077@students.if.itb.ac.id

Saat ini, komunikasi digital semakin dibutuhkan. Komunikasi tersebut dapat berupa *email*, *instant messaging*, forum, *website*, audio, atau video. Fokus tugas akhir ini adalah pengamanan *email* dengan menggunakan algoritma enkripsi. Dengan berbasiskan *corrected block tiny encryption algorithm* yang merupakan algoritma kriptografi kunci simetris, pengguna, baik pengirim maupun penerima, harus memiliki kunci enkripsi yang sama. Tentu saja, orang lebih tertarik untuk memilih kunci yang mudah diingat, tetapi mudah ditebak dibandingkan kunci yang panjang dan sulit diingat. Tugas akhir ini menyediakan sebuah mekanisme untuk membangkitkan kata kunci yang relatif kuat dengan menggunakan masukan dari pengguna.

Tugas akhir ini berisi gambaran singkat mengenai ide awal dan dasar teori pengerjaan tugas akhir, laporan pembangunan perangkat lunak, dan pengujian perangkat lunak. Perangkat lunak ini, atau lebih tepat disebut sebagai *add-on Mozilla Thunderbird*, mampu menangani proses enkripsi dan dekripsi surat elektronik, penghitungan kekuatan kata kunci masukan pengguna, dan pembangkitan kata kunci. Fitur-fitur tersebut dapat membantu pengguna untuk membangkitkan kata kunci hingga mengirimkan surat elektronik yang aman ke penerima. Dalam proses implementasinya, lingkungan implementasi yang digunakan adalah *Windows XP SP3, Mozilla Thunderbird* 2.0.0.21 dengan *add-on DOM Inspector*, dan *Notepad++*. *Add-on* dibangun dengan *XUL (XML User Interface Language)* sebagai pembentuk tampilan dan *Javascript* sebagai bahasa pemrograman. Setelah dilakukan pengujian, semua kebutuhan perangkat lunak berjalan dengan baik, surat elektronik dikirim dalam bentuk terenkripsi, kata kunci bangkitan tergolong relatif kuat, dan enkripsi dan dekripsi berjalan dengan cepat.

Kata kunci: *Corrected block tiny encryption algorithm*, *email*, *Mozilla Thunderbird*, *add-on*.

1 PENDAHULUAN

Sebagai kebutuhan utama dalam bidang komunikasi dan informasi saat ini, teknologi internet mampu menghadirkan informasi dengan cepat dan efisien. Salah satu layanan yang disediakan dengan adanya jaringan internet adalah surat elektronik. Surat elektronik adalah pesan, umumnya berupa teks, yang dikirim dari pengirim kepada penerima melalui jalur internet [BAB08]. Untuk akses dan pengelolaan surat elektronik, hadir *Mozilla Thunderbird* dengan fitur dan berbagai kelebihannya. Salah satunya adalah aplikasi *Mozilla* ini merupakan aplikasi *open source* sehingga tidak perlu khawatir mengenai masalah hak cipta. Berdasarkan data statistik yang ada, aplikasi ini telah diunduh lebih dari 50 juta kali [ASB08]. Jumlah tersebut menunjukkan bahwa *Mozilla Thunderbird* merupakan aplikasi klien surat elektronik yang banyak digunakan orang.

Mozilla Thunderbird menerima surat elektronik baru dengan mengunduhnya setelah melakukan konfirmasi *user* dengan *server* yang berkaitan. Lalu, aplikasi ini mengirimkan surat elektronik yang telah ditulis oleh pengguna dengan menghubungi *server* pengirim surat elektronik dari penyedia jasa. Selain 2 (dua) fungsi utama di atas, *Mozilla Thunderbird* menyediakan fasilitas tambahan *add-ons* yang memungkinkan pengguna untuk menambah atau meningkatkan fitur aplikasi, menggunakan *themes* sesuai keinginan, dan menangani tipe baru suatu konten [MOZ08].

Berikut adalah skema pengirim surat elektronik secara umum:
Surat elektronik ditulis → aplikasi klien surat elektronik (komputer pengirim) → SMTP *server* penyedia layanan surat elektronik pengirim → Internet → POP3 *server* penyedia penyedia layanan surat elektronik penerima → aplikasi klien surat elektronik (komputer penerima) → surat dibaca si penerima [HOW08].

Keamanan data di surat elektronik tidaklah terjamin dan selalu ada resiko terbuka untuk umum, dalam artian semua isinya dapat dibaca oleh orang lain. Hal ini disebabkan oleh karena surat elektronik itu akan melewati banyak *server* sebelum sampai di tujuan. Tidak tertutup kemungkinan ada orang yang menyadap surat elektronik yang dikirimkan tersebut [ITS08]. Untuk mengurangi potensi surat elektronik disadap, surat dapat diamankan dengan menggunakan teknik pengacakan (enkripsi). Pada aplikasi *Mozilla Thunderbird*, tidak disediakan fitur untuk enkripsi surat elektronik. Tetapi melalui *add-ons*, pengguna disediakan sarana untuk mengembangkan fitur enkripsi surat eletronik. Pada tugas akhir ini, *add-on* enkripsi surat elektronik akan dikembangkan dan algoritma yang digunakan adalah *corrected block tiny encryption algorithm*, ke depan disebut sebagai XXTEA.

2 BLOCK CIPHER

Block cipher adalah sebuah tipe algoritma enkripsi kunci simetris yang mentransformasikan sebuah blok *plaintext* berukuran tetap (teks yang tidak terenkripsi) menjadi blok *ciphertext* (teks yang terenkripsi) dengan ukuran yang sama [RSA09]. Transformasi ini berlangsung dengan menggunakan kunci rahasia masukan dari *user*. Proses dekripsi dilakukan dengan menjalankan transformasi kebalikan ke blok *ciphertext* dengan menggunakan kunci rahasia yang sama. Ukuran 1 (satu) blok yang selalu tetap dinamakan *block size* dan untuk sebagian besar *block cipher*, *block size* sebesar 64 *bits*. Sebagai contoh, misalkan terdapat sebuah *plaintext x*. *Plaintext* tersebut dapat dilihat sebagai kumpulan blok:

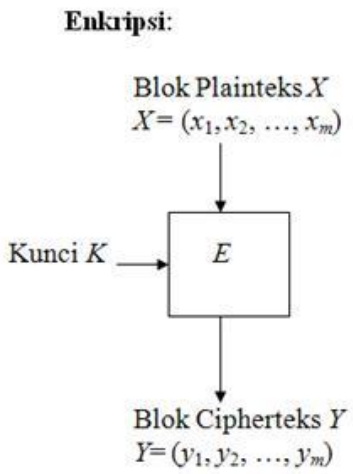
$$x = x_1 x_2 \ldots x_n$$

dimana x_m, $1 \le m \le n$, merupakan satuan blok dari *ciphertext* yang berukuran 64 *bits*.

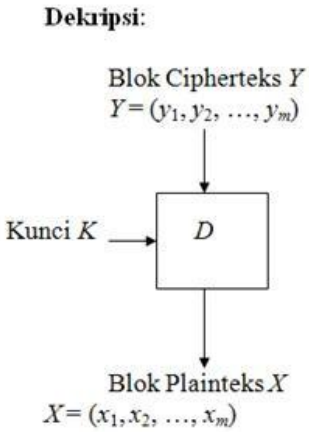
Ketika *user* mengenkripsi *x*, proses enkripsi dilakukan per blok sehingga menghasilkan *ciphertext y* yang dapat dilihat sebagai berikut:

$$y = y_1 y_2 \ldots y_n = e_K(x_1)\ e_K(x_2) \ldots e_K(x_n)$$

Gambar 2-1 dan Gambar 2-2 di bawah menampilkan proses enkripsi dan dekripsi *block cipher* secara umum [MUN06].



Gambar 2-1 Skema Enkripsi *Block Cipher* [MUN06]



Gambar 2-2 Skema Dekripsi *Block Cipher* [MUN06]

Keterangan:

1. Diagram atas menggambarkan proses enkripsi, sedangkan diagram bawah proses dekripsi.
2. E : proses enkripsi dengan masukan *plaintext X* dan kunci *K* yang menghasilkan *ciphertext Y*.
3. D : proses dekripsi dengan masukan *ciphertext Y* dan kunci *K* yang menghasilkan *plaintext X*.

3 *CORRECTED BLOCK TINY ENCRYPTION ALGORITHM*

Pada bab ini, akan dibahas definisi, algoritma penyandian, dan detail sandi yang akan digunakan dalam tugas akhir ini. Secara umum, algoritma yang digunakan adalah *block cipher*.

3.1 Definisi

Corrected Block Tiny Encryption Algorithm, untuk selanjutnya disebut sebagai XXTEA, adalah sebuah algoritma penyandian yang sederhana, tapi kuat yang berbasis iterasi Feistel dan menggunakan banyak ronde untuk mendapatkan keamanan [MOV08]. XXTEA dirancang berupa program kecil yang dapat berjalan pada banyak mesin dan mengenkripsi dengan aman. Algoritma ini menggunakan banyak iterasi dibandingkan program yang rumit sehingga algoritma ini dapat diterjemahkan ke dalam banyak bahasa pemrograman dengan mudah.

XXTEA juga merupakan sebuah algoritma enkripsi efektif yang mirip dengan DES[1] yang dapat digunakan untuk aplikasi web yang membutuhkan keamanan. Ketika menggunakan algoritma ini, sebuah perubahan dari teks asal akan mengubah sekitar setengah dari teks hasil tanpa meninggalkan jejak dimana perubahan berasal.

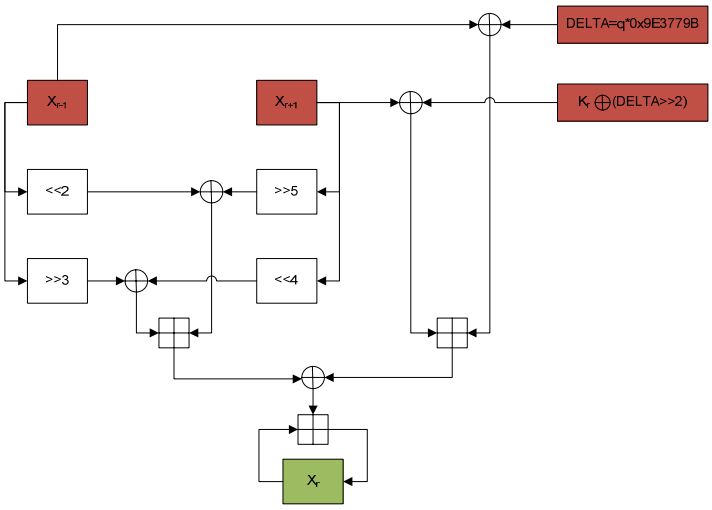
[1] Keterangan lebih lengkap mengenai DES dapat dibaca di http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

XXTEA beroperasi pada blok yang berukuran tetap yang merupakan kelipatan 32 *bits* dengan ukuran minimal 64 *bits*. Jumlah dari putaran lengkap bergantung pada ukuran blok, tetapi terdapat minimal 6 (bertambah terus hingga 32 untuk ukuran blok yang lebih kecil). Algoritma ini menggunakan lebih banyak fungsi pengacakan yang menggunakan kedua blok tetangganya dalam pemrosesan setiap kata dalam blok.

Untuk kemudahan penggunaan dan keamanan secara umum, XXTEA lebih tepat digunakan ketika dapat dipakai untuk kondisi berikut:

1. Perubahan satu *bit* pada *plaintext* akan mengubah sekitar setengah dari total *bits* dari seluruh blok tanpa meninggalkan jejak dimana perubahan dimulai.
2. Walaupun terdapat perubahan yang teratur pada *plaintext* (misalkan nomor pesan), hanya pesan yang sama yang akan memberikan *ciphertext* yang sama dan kebocoran informasi minimal.
3. Jika tidak memungkinkan untuk memasukkan pesan yang panjang, pesan tersebut dapat dipecah menjadi beberapa bagian yang masing-masing berukuran 60 kata.

3.2 Algoritma Penyandian



Gambar 3-1 Satu Iterasi dalam XXTEA

Keterangan simbol:

1. X_r, X_{r-1}, X_{r+1}: blok *plaintext*, dimana r adalah urutan blok yang sedang diacak.

2. *q* : jumlah iterasi yang sedang dilakukan.
3. DELTA : *q* dikalikan dengan konstanta yang bernilai 0x9E3779B.
4. K_r : blok kata kunci ke-*r*, dimana *r* sama dengan keterangan di atas.
5. <<*n* : pergeseran bit ke kiri sebanyak *n* kali.
6. >>*n* : pergeseran bit ke kanan sebanyak *n* kali.
7. ⊕ : operasi XOR.
8. ⊞ : operasi penambahan.

Keterangan warna:
1. Kotak berwarna merah: *input user*.
2. Kotak berwarna hijau: *output* program.

Gambar 3-1 menampilkan proses pengacakan yang dilakukan pada satu iterasinya. Proses iterasi dalam XXTEA dilakukan dalam 2 kali iterasi yang dilakukan secara bersarang. Pada iterasi teratas, iterasi dilakukan sebanyak *q*, dimana:

$$q = 6 + 52/n$$

dengan $n \geq 1$ dimana n adalah jumlah blok dari *plaintext*. Lalu, pada iterasi selanjutnya, iterasi dilakukan sebanyak *n* kali.

Proses pengacakan yang dilakukan dalam satu iterasi XXTEA adalah sebagai berikut:
1. Algoritma akan mengacak blok ke-*r* dari *plaintext*.
2. Proses akan mengambil x_{r-1}, x_{r+1}, DELTA, dan kata kunci sebagai input.
3. Pengacakan pertama: x_{r-1}<<2 di-XOR-kan dengan x_{r+1}>>5.
4. Pengacakan kedua: x_{r-1}>>3 di-XOR-kan dengan x_{r+1}<<4.
5. Hasil yang didapat dari tahap 4 dan 5 ditambahkan.
6. Pengacakan ketiga: x_{r-1} di-XOR-kan dengan *D* yang merupakan perkalian antara konstanta DELTA yang bernilai 0x9E3779B dengan jumlah iterasi pertama yang telah dilakukan.
7. Pengacakan keempat: x_{r+1} di-XOR-kan dengan salah satu blok kata kunci, yaitu blok ke-(*r* XOR *D*>>2).
8. Hasil yang didapat dari tahap ke 6 dan 7 ditambahkan.
9. Hasil yang didapat dari tahap 5 dan 8 di-XOR-kan.
10. Hasil yang didapat pada tahap 9 ditambahkan ke blok *plaintext* ke-*r*.

3.3 Detail Sandi

Tabel 3-1 menjelaskan detail algoritma XXTEA secara singkat mengenai panjang kata kunci, ukuran masing-masing blok, struktur algoritma, dan banyaknya ronde dalam satu proses pengacakan.

Tabel 3-1 Detail Sandi

No	Variabel	Keterangan
1	Panjang kunci	128 *bits*
2	Ukuran blok	Tidak tetap, bergantung pada panjang teks asal
3	Struktur	Jaringan Feistel
4	Ronde	Tidak tetap, 6-32 ronde Feistel (sekitar 3-16 putaran) bergantung pada panjang blok.

4 KATA KUNCI

Pada subbab ini, akan dibahas definisi kata kunci, kekuatan kata kunci, dan cara pengukurannya. Semakin kuat kata kunci yang digunakan dalam enkripsi sebuah *plaintext*, semakin sulit sebuah *ciphertext* untuk dipecahkan. Selain kerumitan algoritma enkripsi yang digunakan, kerumitan kata kunci juga menentukan keamanan *ciphertext* yang didapatkan.

4.1 Definisi

Kata kunci adalah sekumpulan karakter yang dimasukkan seorang pengguna untuk mendapatkan akses ke sumber informasi yang diproteksi [MIC08]. Terdapat sebuah dilema dalam pemilihan dan penggunaan kata kunci, dimana pengguna tidak mudah, atau bahkan tidak mampu mengingat kata kunci yang kuat dan kata kunci yang diingat justru kata kunci yang mudah ditebak.

Kata kunci yang baik harus mengandung karakter campuran atau karakter khusus, dan

tidak boleh mengandung kata-kata yang dapat ditemukan dalam kamus [LAW08]. Kata kunci dapat berupa gabungan huruf kapital atau huruf kecil. Sebuah teknik yang bagus untuk memilih kata kunci adalah dengan menggunakan huruf pertama dari sebuah frase, tapi jangan mengambil frase yang umum diketahui, seperti "*An apple a day keeps the doctor away*" (Aaadktda). Contoh: ambil frase yang bersifat pribadi dan tidak umum digunakan, seperti "*My dog's first name is Rex*" (MdfniR). Selain itu, kata kunci yang baik memiliki panjang minimal 8 (delapan) karakter dan harus mengandung sedikitnya 2 (dua) karakter bukan huruf. Disarankan agar kata kunci selalu diganti minimal sekali sebulan.

4.2 Pengukuran Kekuatan Kata Kunci

Melihat adanya sejumlah saran yang dapat digunakan dalam membangun kata kunci yang baik, kata kunci dapat diukur derajat kekuatannya. Pengukuran kekuatan kata kunci dilakukan dengan melakukan penilaian terhadap kata kunci masukan.

Terdapat sejumlah faktor penilaian yang dapat digunakan [GEE08], yaitu:

1. Panjang karakter kata kunci minimal 8.
2. Gunakan campuran huruf kapital atau huruf kecil.
3. Gunakan minimal 1 angka.
4. Gunakan karakter khusus (!,@,#,$,%,^,&,*,?,_,~).
5. Gunakan prinsip Leet, yaitu penggunaan karakter non-alfabet untuk menggantikan huruf yang memiliki kemiripan yang dekat atau perubahan pengucapan, seperti penggantian huruf "s" yang terakhir dengan "z" atau "(c)ks" untuk "x" [NAT08].

Untuk masing-masing faktor penilaian di atas, diberikan sebuah nilai tambahan jika faktor tersebut dipenuhi. Dengan menggunakan faktor-faktor penilaian di atas, sebuah kata kunci masukan akan dinilai derajat kekuatannya.

5 MEKANISME *ADD-ON* PADA *MOZILLA THUNDERBIRD*

Pada bab 1 telah dijelaskan sebelumnya, *add-on* adalah sebuah fitur yang disediakan pada *Mozilla Thunderbird* untuk memungkinkan *developer* menambah fungsionalitas *Thunderbird*. Dalam pembangunannya, *add-on* dibagi menjadi 2 (dua) bagian penting, yaitu modul antarmuka dan modul *controller*. Pertama, *developer* membangun antarmuka *add-on*. Pembangunan antarmuka menggunakan bahasa XUL.

XUL adalah *Extensible Markup Language* (XML) *grammar* yang menyediakan komponen dasar antarmuka, seperti *button, menu, toolbar, tree*, dan lain-lain. Aksi user yang dikaitkan dengan fungsionalitas dibangun dengan menggunakan *Javascript*. Untuk mengembangkan *Thunderbird*, pengembang memodifikasi antarmuka *Thunderbird* dengan menambahkan atau memodifikasi komponen dasarnya. Pengembang menambahkan komponen dasar dengan memasukan elemen DOM XUL ke *window Thunderbird* dan memodifikasinya dengan menggunakan *scripts* dan *event handlers* yang sesuai [DEV09]. *Scripts* dan *event handlers* dapat dibangun dengan menggunakan bahasa C dan *Javascript*. Pada tugas akhir ini, *scripts* dan *event handlers* dibangun dengan *Javascript*.

6 ANALISIS MASALAH

Bab ini akan membahas masalah-masalah yang ditemui dalam pengembangan perangkat lunak dan bagaimana menangani masalah tersebut.

6.1 Masalah Kekuatan Kata Kunci

Salah satu permasalahan yang dapat muncul dalam pembuatan tugas akhir ini adalah kekuatan kata kunci yang digunakan oleh *user*. Pada subbab 4.2 Pengukuran Kekuatan Kata Kunci, terdapat sejumlah kriteria yang dapat digunakan untuk mengukur kekuatan kata kunci, tetapi tidak semua orang mengetahui kriteria tersebut sehingga kata kunci yang digunakan cenderung lemah. Misalkan *user* memilih kata-kata umum, seperti "123", "*password*", atau "*qwerty*", *ciphertext* dapat

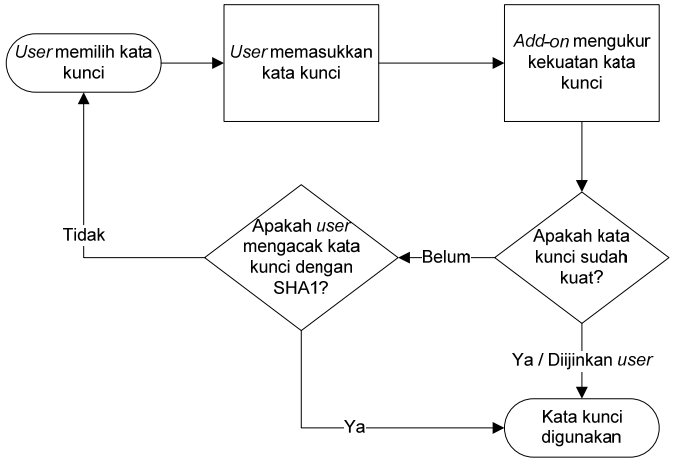
mudah diserang dengan menggunakan *dictionary attack*. Selain itu, kriptanalis tidak perlu mengkhawatirkan salah memasukkan kata kunci sebab tidak adanya *counter* untuk kata kunci yang salah dimasukkan, sehingga *user* harus menggunakan kata kunci yang kuat agar kata kunci sulit ditebak dan *ciphertext* sulit untuk dipecahkan.

Cara yang akan digunakan dalam menangani pemilihan kata kunci yang kuat yang dipilih oleh *user* adalah dengan menambahkan sebuah indikator yang menunjukkan seberapa kuat kata kunci yang digunakan. Terdapat sejumlah kriteria yang dapat digunakan untuk mengukur kata kunci. Berikut adalah rumus yang akan digunakan dalam penghitungan kekuatan kata kunci:

```
T = PKK + HRF + ANG + SC + COM
```

Keterangan:
1. *T* : total nilai kekuatan kata kunci dengan nilai maksimal 100. Total nilai yang diperbolehkan sebagai kata kunci adalah 60 ke atas.
2. *PKK* : faktor panjang kata kunci.
3. *HRF* : faktor jumlah dan kombinasi huruf.
4. *ANG* : faktor jumlah angka.
5. *SC* : faktor *special character*.
6. *COM* : faktor kombinasi dari kemunculan huruf, angka dan *special character*.

Jika kekuatan kata kunci yang dimasukkan *user* sudah cukup aman, *add-on* tidak akan memberikan peringatan kepada *user* mengenai kata kunci tersebut, tetapi jika kata kunci masukan *user* tidak aman, *add-on* dapat membantu *user* dalam memilih. Untuk memudahkan user, *add-on* menyediakan fasilitas pembangkitan kata kunci dimana proses pembangkitan dapat dilakukan setelah *user* memasukan sebuah kata kunci. Jika *user* merasa kesulitan dalam memilih kata kunci, *user* dapat memerintahkan *add-on* agar kata kunci disandi terlebih dahulu dengan menggunakan algoritma SHA1 yang kemudian di-*encode* dengan Base64. Hal ini dilakukan agar kata kunci lebih acak, sehingga lebih sulit untuk ditebak.



Gambar 6-1 Skema Pemilihan Kata Kunci

6.2 Masalah Implementasi XXTEA

Bahasa yang digunakan dalam pembangunan *add-on Mozilla Thunderbird* menggunakan *Javascript*. Lalu berdasarkan bab 3 **Error! Reference source not found.**, XXTEA adalah algoritma *block cipher*. Karena proses enkripsi dilakukan per blok, *string* masukan harus dikonversi terlebih dahulu menjadi bentuk *array of long*[2] yang berukuran 64 *bits* agar mendapatkan ukuran blok yang tetap 64 *bits* dan memenuhi syarat penggunaan XXTEA.

Proses konversi string menjadi sebuah *array of long* dan proses sebaliknya dilakukan dengan menggunakan pergeseran bit ke kiri (*left shift register* - LSR) dan ke kanan (*right shift register* - RSR). Tetapi pada *Javascript*, sebuah variabel hanya dapat digeser dengan LSR dan RSR maksimal sebanyak 32 kali, sedangkan proses konversi ini membutuhkan 64 kali. Oleh karena itu, *block size* yang digunakan adalah 32 *bits*. Hal ini tidak akan mengganggu proses enkripsi dan dekripsi sebab *block size* yang digunakan merupakan kelipatan 32 *bits*.

Setelah *plaintext* dienkripsi menjadi *ciphertext*, *ciphertext* yang masih dalam bentuk *array of long* dikonversi kembali menjadi *string*. Hal yang sama akan dilakukan

[2] Ukuran *byte* dari tipe data *long* dapat dilihat di referensi [EDU09].

ketika melakukan proses dekripsi. *Ciphertext* akan dikonversi terlebih dahulu menjadi *array of long*, lalu *ciphertext* akan didekripsi. Proses konversi *string* menjadi *array of long* akan menyelesaikan masalah enkripsi blok *plaintext* yang tidak tepat 32 *bits*. Proses konversi ini akan menganggap karakter kosong sebagai nilai 0.

6.3 Masalah Encoding Ciphertext

Karena proses enkripsi *plaintext* dilakukan ketika berbentuk *array of long* dengan tujuan untuk mempermudah proses enkripsi, *ciphertext* yang telah dikonversi kembali ke dalam bentuk *string* akan menghasilkan *string* yang dapat menimbulkan *unicode conversion problem*, dimana nomor *unicode*-nya tidak didukung oleh MIME. Untuk mencegah masalah kompatibilitas ini, *ciphertext* tersebut akan di-*encode* terlebih dahulu menjadi Base64 sebelum ditampilkan ke *user*, sehingga surat elektronik yang telah dienkripsi akan dalam *encoding* Base64. Proses dekripsi *ciphertext* akan melakukan proses kebalikannya, yaitu proses *decoding* terlebih dahulu hingga akhirnya dilakukan dekripsi.

6.4 Masalah Kenyamanan Add-on

Add-on yang dibangun sebaiknya nyaman digunakan. Pada tugas akhir ini, hal tersebut dapat diukur dengan menghitung jumlah langkah tambahan untuk proses pengiriman dan tampilan *add-on* yang mudah dimengerti. Dengan menggunakan pertimbangan tersebut, dapat disimpulkan bahwa *add-on* dikatakan nyaman jika jumlah langkah tambahan relatif sedikit dan *icon* yang digunakan menggambarkan proses yang dijalankan.

Pada tugas akhir ini, jumlah langkah tambahan dalam proses pengiriman sekitar 2 (dua) hingga 3 (tiga) langkah, yaitu

1. *User* memasukkan kata kunci.
2. *User* menambah kekuatan kata kunci (langkah opsional).
3. *User* mengenkripsi surat elektronik.

Penggunaan *icon* pada *add-on* harus merepresentasikan proses *add-on* dengan baik. Misalkan *icon* untuk proses enkripsi berupa gambar sebuah surat yang berubah menjadi surat yang diacak dengan simbol gembok.

7 HASIL DAN PENGUJIAN

Setelah melihat hasil pengujian, dapat dievaluasi sebagai berikut:

1. Fungsionalitas program berjalan dengan baik.
2. Walaupun pesan terenkripsi dimiliki oleh pihak ketiga dan pihak ketiga tidak mengetahui kata kunci yang digunakan, mereka akan kesulitan mengetahui isi pesan sebab dengan kata kunci masukan yang mirip dengan kata kunci yang benar, pesan terdekripsi tidak memiliki makna. Hanya dengan kata kunci yang benar, pesan terdekripsi dapat mudah dibaca.
3. Pesan terenkripsi yang dikirimkan dari mesin pengguna memang dalam bentuk *ciphertext*. Sehingga ketika pengguna menghubungi sebuah *mail server* dalam *unsecure connection* dan terdapat seorang pihak ketiga yang menyadap isi pesan, pihak ketiga tersebut mendapatkan *ciphertext* yang hanya dapat dibuka dengan kunci yang sesuai.
4. Pembangkitan kata kunci dari kata kunci masukan yang kurang kuat memberikan kata kunci keluaran yang relatif kuat. Skor kata kunci keluaran berkisar 66 hingga 76. Hal yang sama terjadi jika kata kunci masukan tergolong relatif kuat. Hasil skor kata kunci yang didapat berkisar 66 hingga 76 yang berarti pembangkitan kata kunci pada beberapa kasus justru tidak meningkatkan kekuatan kata kunci.
5. Kecepatan enkripsi dengan XXTEA tergolong cepat hal ini dapat dilihat dari waktu enkripsi yang sangat singkat, yaitu antara 11-16 ms untuk mengenkripsi sekitar 1000 hingga 2000 *bytes* dan 31-32 ms untuk mengenkripsi sekitar 3000 hingga 4000 *bytes* pada lingkungan pengujian. Hasilnya dapat dilihat pada Tabel 7-1 Waktu Enkripsi (dalam milidetik).

Tabel 7-1 Waktu Enkripsi (dalam milidetik)

Ukuran Pesan\Panjang Kata Kunci	4 Char	8 Char	12 Char	16 Char
1000 bytes	0	15	16	16
2000 bytes	15	15	16	16
3000 bytes	31	31	32	32
4000 bytes	31	32	31	31

8 KESIMPULAN

Kesimpulan yang dapat diambil selama pengerjaan tugas akhir ini sebagai berikut:

1. *Add-on* yang mampu melakukan enkripsi, dekripsi, penghitungan skor kata kunci, dan pembangkitan kata kunci dengan menggunakan XXTEA dapat dibangun.
2. Dalam membangun sebuah program kriptografis berbasiskan algoritma kunci simetris, pemilihan kata kunci adalah hal yang penting agar pesan terenkripsi sulit diserang dengan *dictionary attack*.
3. Dengan menggunakan rumus penghitungan skor dan pembangkitan kata kunci pada tugas akhir ini, skor yang didapat cenderung sama. Hal ini berarti kata kunci keluaran cenderung memiliki pola yang sama, yaitu panjang kata kunci 40 karakter dalam *encoding* Base64 sehingga memudahkan pihak ketiga untuk menebak kata kunci yang digunakan.
4. Dalam pembangunan *add-on Mozilla Thunderbird*, *developer* akan menemui kesulitan dalam mencari dokumentasi yang lengkap mengenai struktur XUL dan fungsi *Javascript* yang dapat digunakan.

DAFTAR REFERENSI

[ASB08] http://tb.asbjorn.it/pages/dlgraph.php, waktu akses: 24 September 2008, 21:00. Durasi: 30 menit.

[BAB08] http://dictionary.babylon.com/e-mail, waktu akses: 24 September 2008, 20:00. Durasi: 30 menit.

[DEV09] https://developer.mozilla.org/en/Building_a_Thunderbird_extension, waktu akses: 20 Agustus 2009 pukul 12:00 WIB. Durasi: 1 jam.

[EDU09] http://www.educator.com/quicknotes/computer-science/javascript/primitive-data-types.html, waktu akses: 27 Februari 2009 pukul 10:30 WIB. Durasi: 1 jam.

[GEE08] http://www.geekwisdom.com/dyn/passwdmeter, waktu akses: 6 Desember 2008 pukul 15:00 WIB. Durasi: 1 jam.

[HOW08] http://communication.howstuffworks.com/email.htm, waktu akses: 24 September 2008, 20:30. Durasi: 30 menit.

[ITS08] http://www.itsecurity.com/features/five-steps-email-security-092106/, waktu akses: 24 September 2008, 20:30. Durasi 30 menit.

[LAW08] http://download.lawr.ucdavis.edu/pub/it/CambridgePWStudy.pdf, waktu akses: 6 Desember 2008 pukul 14:00 WIB.

[MIC08] http://www.micro2000.co.uk/network_glossary.htm, waktu akses: 6 Desember 2008 pukul 14:00 WIB. Durasi: 1 jam.

[MOV08] http://www.movable-type.co.uk/scripts/xxtea.pdf, waktu akses: 11 September 2008, 10:00. Durasi: 2 jam.

[MOZ08] http://developer.mozilla.org/en/Extensions, waktu akses: 23 September 2008, 20:00. Durasi: 30 menit.

[MUN06] Munir, Rinaldi. 2006. Diktat Kuliah IF5054 Kriptografi.

[RSA09] http://www.rsa.com/rsalabs/node.asp?id=2168, waktu akses: 27 Februari 2009 pukul 09:30 WIB. Durasi: 1 jam.