Penerapan Kriptografi dalam Program Enkripsi Sederhana JLBEncrypt

Jeffrey Lingga Binangkit 13512059
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13512059@std.stei.itb.ac.id
jeffrey.lingga@s.itb.ac.id

***Abstrak*—Kriptografi merupakan salah satu bab yang dipelajari dalam Matematika Diskrit. Kriptografi merupakan keilmuan untuk menjaga keamanan pesan. Penerapan kriptografi dapat dilakukan di pelbagai bidang kehidupan, mulai dari komunikasi rahasia oleh pemerintahan sampai berkirim pesan lewat surel. Salah satu penerapan kriptografi adalah enkripsi data di komputer. JLBEncrypt versi pertama yang saya buat adalah salah satu enkripsi sederhana dalam mengubah file menjadi *chipertext*, sehingga tidak dapat dibuka oleh orang yang tidak memiliki kunci. JLBEncrypt sendiri menggunakan kunci berbentuk *string* untuk mengenkripsi dan mendekripsi data.**

***Index Terms*—Algoritma, Dekripsi, Enkripsi, JLBEncrypt.**

I. PENDAHULUAN

Dunia semakin berkembang, jaman makin modern, sekat-sekat antar ruang semakin berkurang, dan privasi menjadi barang mahal di saat ini. Padahal, setiap orang selalu memiliki kepentingan yang tidak boleh diketahui orang lain, seperti informasi yang dikirimkan antar kepala negara, PIN dalam transaksi perbankan, atau sekadar password untuk login di social media. Pencurian data dapat terjadi mulai dari orang-orang biasa sampai kepada penyadapan para petinggi negara.

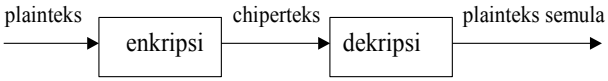
Pesan maupun data akan lebih aman jika dikodekan terlebih dahulu sebelum dikirim, disimpan, atau ketika akan membuang perangkat, sehingga terjaga kerahasiaannya sebelum mencapai tujuan. Setelah mencapai tujuan atau ketika ingin membuka lagi suatu data yang dikodekan sebelumnya, seseorang harus mampu tahu bagaimana mendekodekan suatu pesan, sehingga dapat tersampaikan.

Bukan hanya pesan, data penting di perangkat milik sendiri pun tidak luput dari incaran para jahil yang siap untuk mengintip file-file seseorang, jika orang tersebut tidak berhati-hati dalam menyimpan data-data penting.

Karena itu saat ini banyak terdapat aplikasi yang digunakan untuk mengkodekan pesan atau data sebelum dikirimkan ke tujuan, kemudian dikembalikan ke wujud aslinya sehingga pesan tersampaikan dengan aman. Salah satu aplikasi yang dapat digunakan adalah JLBEncrypt, aplikasi yang saya buat sendiri untuk mengenkripsi data-data penting/pribadi yang sering terlupakan oleh saya seperti PIN ATM, password situs-situs tertentu serta data-data lain. JLBEncrypt dapat diunduh secara gratis di repositori mediafire berikut ini
https://www.mediafire.com/folder/tqgu8gd7lgxx3/JLBEncrypt%20Project%20(MATDIS)

II. CHIPERTEXT, ENKRIPSI DAN DEKRIPSI

Chipertext adalah pesan yang telah disandikan sehingga tidak memiliki makna lagi[1]. Tujuannya agar pesan tidak dapat dimengerti maknanya oleh pihak lain. *chipertext* sejatinya merupakan pesan asli yang disamarkan, sehingga pihak-pihak perantara pesan tidak dapat membaca isi pesan tersebut. Pesan yang dikodekan ini harus dapat dikembalikan lagi ke bentuk semula agar dapat dibaca oleh tujuan pesan tersebut.



Gambar 1. *plaintext* dan *chipertext* [1]

Chipertext diperoleh dari *plaintext*, yaitu pesan awal yang masih mentah, kemudian melalui sebuah tahapan yaitu enkripsi. Dalam enkripsi ini, pesan dikenai fungsi matematika yang dibangkitkan oleh kunci-kunci tertentu yang dimasukkan oleh pengirim pesan. Memasukkan kunci yang berbeda dapat menghasilkan *chipertext* yang berbeda, meski pada awalnya *plaintext* yang dikodekan sama. Semua ini bergantung juga pada kunci yang dimasukkan oleh pengirim, karena dari kuncilah fungsi memiliki nilai untuk kemudian digunakan untuk merombak pesan mentah.

Jika fungsi yang dibangkitkan oleh proses enkripsi dilambangkan dengan E, *plaintext* dilambangkan dengan P dan *chipertext* dilambangkan dengan C, maka didapat persamaan sebagai berikut

$$E(P) = C$$

Setelah menjadi *chipertext*, pesan kehilangan status rahasianya. Pesan boleh-boleh saja dibaca oleh orang lain, dengan asumsi bahwa pesan tidak dapat lagi dipahami oleh siapapun tanpa mengembalikan ke bentuk *plaintext* lagi. Asumsi ini akan menjadi benar jika enkripsi dilakukan dengan baik, yang artinya algoritma enkripsi kuat.

Chipertext kemudian dikirimkan ke tujuan. Setelah diterima oleh tujuan pesan, orang tersebut harus memiliki kunci yang dapat digunakan untuk membangkitkan fungsi Dekripsi, sehingga pesan yang tadinya telah menjadi *chipertext* dapat kembali terbaca sebagai *plaintext*.

Jika fungsi yang dibangkitkan oleh proses dekripsi dilambangkan dengan D, *plaintext* dilambangkan dengan P dan *chipertext* dilambangkan dengan C, maka didapat persamaan sebagai berikut.

$$D(C) = P$$

Karena proses enkripsi kemudian dekripsi mengembalikan pesan ke wujud semula, maka kedua fungsi, yakni enkripsi (E) dan dekripsi (D) harus memenuhi persamaan berikut.

$$D(E(P)) = P$$

III. KRIPTOGRAFI

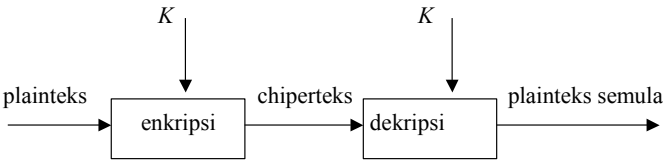
> Algoritma kriptografi -atau *chiper*- adalah fungsi matematika yang digunakan untuk enkripsi dan dekripsi.
>
> -Rinaldi Munir, Diktat Kuliah IF2120

Gambar 2
Salah satu mesin cipher, German Lorenz Cipher Machine, dipakai di Perang Dunia II
sumber :
http://www.codesandciphers.org.uk/lorenz/pictures/lorenz1.jpg

Kriptografi (atau Kriptologi; berasal dari bahasa Yunani κρυπτός, "tersembuny, rahasia"; dan γράφειν, graphein, "tulisan", atau -λογία, -logia, "ilmu"[2] adalah ilmu yang mempelajari teknik untuk mengamankan komunikasi dari adanya pihak ketiga (adversaries)[3]

Kriptografi sudah mulai digunakan sejak tahun 400 SM, oleh tentara-tentara Sparta. Pada zaman tersebut, para tentara saling berkirim pesan dengan menggunakan alat yang disebut dengan *scytale*. Fungsi enkripsi dan dekripsi yang digunakan pada jaman tersebut masih sangat sederhana, yakni dengan menggeser tiap huruf alfabet ke kanan 3 slot, dan ketika mencapai huruf terakhir, kembali ke awal (huruf a). Enkripsi model ini merupakan metode enkripsi tertua. Dapat dituliskan sebagai berikut :

Enkripsi : $c = E(p) = (p+K) \bmod 26$
Dekripsi : $p = D(c) = (c-K) \bmod 26$



Gambar 3. Skema Kunci Enkripsi dan Dekripsi [1]

Seperti yang terlihat pada persamaan di atas, fungsi dibangkitkan oleh kunci yang digunakan untuk transformasi *enchipering* dan *dechipering* (dalam hal ini kuncinya merupakan K=3). Algoritma kriptografi dapat tidak rahasia namun kunci ini bersifat rahasia.

Dalam ilmu kriptografi dikenal dua istilah : Kriptografer dan Kriptanalis. Kriptografer menggunakan enkripsi untuk merahasiakan pesan dan mendeskripsikannya kembali. Ia menerjemahkan *chipertext* menjadi *plaintext* atas legitimasi pengirim atau penerima pesan. Kriptanalis mempelajari metode enkripsi dan *chipertext* guna menemukan *plaintextnya*. Kriptanalis cenderung bekerjasama dengan penyusup yang sebenarnya tidak berhak mengetahui informasi yang terdapat pada *chipertext*[1].

IV. JLBENCRYPT : PENERAPAN KRIPTOGRAFI DALAM PENGAMANAN FILE

Bukan rahasia lagi bahwa semakin berkembangnya teknologi, memang memudahkan dalam saling berkirim pesan, bertukar data maupun hal-hal lain. Namun ini juga berarti semakin mudah bagi pencuri data.

Salah satu cara untuk mengamankan data dari tangan-tangan jahil adalah dengan mengkodekannya. Di sini dapat diterapkan pula kriptografi, yang awalnya hanya dalam saling bertukar pesan, dapat pula menjadi sebuah metode untuk menyamarkan data-data yang ada di komputer kita sendiri.

JLBEncrypt merupakan salah satu solusi untuk melindungi data-data penting di komputer dari tangan-tangan jahil. Aplikasi JLBEncrypt sangat ringan dan cukup kuat untuk sekadar menyamarkan data-data di komputer.

JLBEncrypt mengaplikasikan kriptografi dalam menyamarkan data dengan menggunakan kunci berupa string. JLBEncrypt membangkitkan nilai fungsi berbeda untuk string yang berbeda, sehingga cukup kuat.

Enkripsi JLBEncrypt dilakukan dengan cara membaca file pertama yang ingin dienkripsi. Program kemudian meminta sebuah masukan string untuk menjadi kunci enkripsi. Penggunaan string didasari oleh banyaknya kombinasi string yang bisa diciptakan.

User kemudian diminta untuk memasukkan nama file kedua, yang akan menjadi nama file terenkripsi. File ini kemudian memuat hasil enkripsi dari file pertama. String yang pertama kali dimasukkan oleh pembuat akan digunakan untuk mentransformasikan file.

Dalam versi pertama, JLBEncrypt memanfaatkan potensi dari penggunaan string. Fungsi enkripsi yang digunakan adalah sebagai berikut :

E(Fn, F, K) = C = R(K) & T(Fn) & S(F)

Dimana E merupakan fungsi enkripsi yang nilainya dibangkitkan oleh tiga parameter yaitu Fn, filename dari data yang akan dienkripsi, kemudian F, yaitu file itu sendiri, dan K, kunci enkripsi. C merupakan *chipertext* yang tercipta dari enkripsi file tersebut. C tersusun atas empat bagian penting yang disambungkan menjadi satu file (di sini dilambangkan dengan &). Masing-masing bagian terbentuk dari fungsi-fungsi tersendiri.

Diawali header yang berupa "JLBEncrypted>" pada versi pertama, dimulailah sebuah file JLBEncrypt. Kemudian R merupakan fungsi yang mentranformasi K yaitu kunci. Kunci ditransformasikan karena kemudian kunci dikodekan pula ke dalam file. K perlu dikodekan ke dalam file untuk memperkuat keamanan file tersebut, karena ketika seseorang mencoba untuk membuka file, ia akan kesulitan karena kunci secara sepesifik telah diinjeksi ke file, sehingga sulit untuk coba-coba. T merupakan fungsi yang mengkodekan filename. Filename sewajarnya dikodekan ke dalam file, karena file kurang berguna pula jika tidak diketahui namanya, terutama ekstensi file tersebut. Jika ketika mengembalikan file ke bentuk pesan sang penerima tidak mengetahui nama filenya, akan sulit untuk menggunakan sewajarnya di komputer. Yang terakhir adalah S, fungsi yang mengkodekan file. Fungsi ini dibangkitkan secara spesifik oleh kunci tertentu, sehingga sulit untuk menerka-nerka. Bagian paling penting dari pengkodean *chipertext* ada di sini, karena bagian file inilah yang memuat data penting anda.

Untuk menerjemahkan *chipertext* hasil enkripsi JLBEncrypt, digunakan fungsi sebagai berikut :

D(C, K) = P = Fn, F

Dimana D merupakan fungsi dekripsi yang nilainya dibangkitkan oleh *chipertext* dan kunci spesifik. Mengingat kunci tertulis secara tersamar oleh fungsi R di file C, jika kunci tidak match, maka *plaintext*, yaitu P, tidak dapat dihasilkan. Karena dalam konteks file komputer, di sini P dibagi menjadi Filename (Fn) dan File itu sendiri (F). Fn dibangkitkan dari *inverse* fungsi T, serta File F dibangkitkan dari *inverse* fungsi S. Dekripsi melakukan pengecekan header dari file, begitu pula key, sementara filename didekripsi dan diletakkan di file. Hasil akhir dari dekripsi sebuah *chipertext* C adalah sebuah file lengkap dengan filenamenya.

V. JLBENCRYPT V1 : ALGORITMA ENKRIPSI

Setiap aplikasi pasti didasari oleh cara berpikir tertentu yang dapat dituangkan dalam algoritma. Begitu pula JLBEncrypt yang saya buat berdasarkan beberapa algoritma sederhana yang mangkus, maksimal berorde n (O(n)), untuk menyamarkan file.

Seperti yang telah dibahas di bagian sebelumnya, enkripsi oleh aplikasi JLBEncrypt mengikuti fungsi enkripsi berikut :

E(Fn, F, K) = C = R(K) & T(Fn) & S(F)

Pertama, fungsi R. Fungsi ini pada dasarnya melakukan transformasi terhadap kunci K yang dimasukkan si pengirim pesan. Kunci ditransformasi berdasarkan algoritma tertentu kemudian ditempatkan sedemikian rupa, sehingga di kemudian hari ketika file didekripsi dengan kunci yang salah, pesan tidak bisa dibuka. Berikut adalah fungsi R :

R(K) : K[i] ← K[i]+2*i

Di fungsi R, karakter kunci K ke-i, dengan i dari 0 sampai panjang K dikurangi satu, ditambahkan dengan nilai dua kali i. Setelah melalui fungsi ini, kunci K akan disamarkan. Contohnya adalah kunci "kita" akan berubah menjadi "kkxg".

Kemudian fungsi T yang berfungsi untuk mengkodekan nama file. Dalam enkripsi JLBEncrypt, filename diletakkan di dalam file terenkripsi sehingga ketika file didekripsi dari file, sudah memiliki nama file.

T(Fn) : j ← i mod (*length*(K)-1)
Fn[i] ← Fn[i]-3*K[j]

Di sini fungsi T menyamarkan nama file (Fn) dengan beberapa transformasi cukup kuat, dimana nama file ditambahkan dengan karakter tertentu yang berada di dalam kunci K. Pengkodean *filename* menjadi penting karena file yang baik memiliki nama yang dapat menggambarkan file tersebut, sementara kita ingin menyamarkan file, sehingga nama harus disamarkan sebaik mungkin. Fungsi T menambahkan nilai tiga dikali karakter kunci K yang ke-j dimana j adalah i mod panjang karakter K dikurang satu, sehingga nilai j berkisar di antara nol dan panjang elemen K dikurangi satu.

Yang terakhir adalah fungsi S, yang digunakan untuk menyamarkan file F. File F ditransformasikan sedemikian rupa sehingga tidak menggambarkan file aslinya lagi, kemudian dituliskan ke file tujuan. Fungsi S dapat dijabarkan sebagai berikut :

S(F) : F[i] ← F[i]+*length*(K)+K[i mod (*length*(K)-1]

Di sini fungsi S menambahkan nilai panjang karakter kunci K dan karakter kunci K ke-i mod panjang K dikurang satu. Fungsi enkripsi file menjadi yang paling rumit. Sebabnya adalah file inilah bagian paling penting untuk dienkripsi. Sebisa mungkin nama file masih boleh diterka-terka, namun file harus sulit untuk ditebak. Di sini

fungsi S mentransformasikan F yang dilambangkan sebagai kumpulan elemen F[i].

Setelah seluruh bagian diproses dan telah terenkripsi, yang terakhir dilakukan oleh fungsi E adalah menyatukan semuanya ke dalam struktur file *chipertext* C. Sehingga secara keseluruhan dapat dituliskan sebagai berikut

E(Fn, F, K) :

. . . {melakukan fungsi R, T dan S}

C.H ← JLBEncrypt>
C.eK ← K
C.eFn ← Fn
C.eF ← F

VI. JLBENCRYPT V1 : ALGORITMA DEKRIPSI

Setelah dibahas algoritma untuk enkripsi, di bagian ini akan dibahas mendetail mengenai algoritma dekripsi JLBEncrypt.

Seperti yang telah dibahas di bagian sebelumnya, enkripsi oleh aplikasi JLBEncrypt mengikuti fungsi enkripsi berikut :

D(C, K) = P = Fn, F

Karena dalam konteks file komputer, di sini P dibagi menjadi Filename (Fn) dan File itu sendiri (F). Fn dibangkitkan dari balikan (*inverse*) fungsi T, serta File F dibangkitkan dari balikan fungsi S. Dekripsi melakukan pengecekan header dari file, begitu pula kunci, sementara nama file didekripsi dan diletakkan di file tujuan. Hasil akhir dari dekripsi sebuah *chipertext* C adalah sebuah file yang telah lengkap dengan *filename*-nya.

Pertama adalah balikan fungsi R yang saya tuliskan sebagai fungsi r. Fungsi r memiliki satu parameter saja yaitu *chipertext* C. Namun di sini C memiliki bagian-bagian di dalamnya yaitu *header* (H), *encrypted key* (eK), *encrypted filename* (eFn) dan *encrtypted file* (eF) yang coba dituliskan seperti seakan-akan sebuah tipe data terstruktur. Berikut ini fungsi r :

r(C) : Kt[i] ← C.eK[i]
K[i] ← C.eK[i]-2*i

Nilai kunci C.eK disimpan di Kt, kemudian coba untuk dikembalikan ke semula dan disimpan di string K. Kunci perlu dikembalikan ke semula di K karena program perlu untuk mengecek apakah kunci yang dimasukkan oleh orang yang menerima pesan ini benar atau tidak. Di lain sisi, untuk nilai kunci yang awal di C.eK akan digunakan kembali dalam dekripsi file serta *filename*, sehingga harus disimpan di Kt.

Fungsi berikutnya adalah balikan dari T, yang dituliskan sebagai t. Sama seperti r, fungsi t hanya perlu menerima *chipertext* C yang memiliki struktur tertentu. Namun fungsi t dan sesudahnya (fungsi s) hanya akan dijalankan jika nilai K hasil fungsi r cocok dengan kunci masukan dari user. Fungsi t dapat dituliskan sebagai berikut :

t(C) : j ← i mod (*length*(Kt)-1)
Fn[i] ← C.eFn[i]+3*Kt[j]

Di sini Kt merupakan kunci terenkripsi yang telah disimpan oleh fungsi f(C). Nama file dicatat oleh Fn. Fn sendiri mencatat C.eFn setelah ditransformasi balik ke bentuk asalnya dengan cara menambahkan tiga kali elemen kunci Kt ke-j, dimana j adalah i mod panjang Kt dikurangi satu. Nilai i dimulai dari nol sampai dengan panjang Fn dikurang satu.

Bagian terakhir yang memiliki keamanan paling tinggi adalah balikan fungsi S yang saya notasikan sebagai fungsi s. Fungsi s digunakan untuk memprosesbalikkan C.eF yang merupakan file terenkripsi dalam *chipertext* C. Sama seperti fungsi r dan t, fungsi s memerlukan satu parameter untuk mendekripsi yaitu *chipertext* C. Fungsi s dapat dituliskan sebagai berikut :

s(C) : j ← i mod (*length*(Kt)-1)
F[i] ← C.eF[i]-*length*(Kt)-Kt[j]

Di sini file F dimunculkan dengan menggunakan fungsi s. Fungsi s mengassign elemen C.eF ke-i dikurangi panjang key Kt dikurangi lagi dengan elemen Kt ke-j. Nilai j di sini adalah i mod panjang karakter Kt dikurangi satu. Nilai i berkisar di antara nol dan panjang file terenkripsi (C.eF) yang merupakan bagian dari struktur C.

Sementara bagian header tidak diproses sama sekali, karena hanya berlaku sebagai mark.

Setelah seluruh algoritma dekripsi berjalan dengan baik, maka telah tercipta file F dan filename Fn. Keduanya kemudian disatukan menjadi *chipertext* P lengkap dengan file dan filenamenya. Dapat dituliskan sebagai berikut.

D(C) :

. . . {melakukan fungsi r, t dan s}

P.Fn ← Fn
P.F ← F

VII. KESIMPULAN

Saat ini diperlukan peningkatan keamanan dalam berkirim pesan dan menyimpan data. Salah satu cara mengamankan data serta pesan adalah dengan menggunakan ilmu kriptografi. Kriptografi merupakan ilmu untuk menjaga keamanan pesan, namun kemudian dapat dikembangkan dalam enkripsi data. Algoritma kriptografi adalah fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Sebuah informasi berupa *plaintext* dikirimkan oleh seseorang setelah melalui enkripsi sehingga berubah menjadi *chipertext*. Penerima harus melakukan dekripsi untuk mendapat pesan sebenarnya. Kedua kegiatan enkripsi dan dekripsi ini membutuhkan kunci untuk dapat berjalan. Salah satu penerapan kriptografi dalam kehidupan adalah dengan pengaplikasian enkripsi-dekripsi terhadap file-file yang ada di komputer. Salah satu aplikasi sederhana untuk mengamankan data adalah menggunakan JLBEncrypt

yang saya buat sendiri. Algoritma enkripsi dalam JLBEncrypt dapat dirumuskan sebagai E(Fn, F, K) = C = R(K) & T(Fn) & S(F) sementara untuk dekripsi dapat dituliskan sebagai D(C, K) = P = Fn, F.

REFERENSI

[1] Rinaldi Munir, Diktat Kuliah IF2120 Matematika Diskrit. Bandung : Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung, 2006, ch. 5.

[2] Liddell and Scott's Greek-English Lexicon. Oxford University Press. 1984.

[3] Rivest, Ronald L. "Cryptology". In J. Van Leeuwen. *Handbook of Theoretical Computer Science* **1**. Elsevier. 1990.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 14 Desember 2013

ttd

Jeffrey Lingga Binangkit
13512059