ALGORITMA *BRUTE-FORCE* DALAM MEMFAKTORKAN BILANGAN *RSA*

Amudi Sebastian[1], Sonny Indro Prabowo[2], Febrian Setiadi[3]

Laboratorium Ilmu dan Rekayasa Komputasi
Departemen Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung

E-mail : if13015@students.if.itb.ac.id[1], if13017@students.if.itb.ac.id[2], if13028@students.if.itb.ac.id[3],

Abstrak

Kriptografi adalah ilmu sekaligus seni untuk menjaga kerahasiaan pesan (informasi atau data) dengan cara menyamarkannya (*to crypt* artinya menyamar) menjadi bentuk tersandi yang tidak mempunyai makna. Sekarang ini kerahasiaan informasi menjadi sesuatu yang penting. Informasi yang rahasia perlu disembunyikan agar tidak diketahui oleh orang yang tidak berhak. Kini kriptografi tengah di terapkan dalam pengiriman sms (*short message service*),karena kini telah ada transaksi bank dengan sms (*sms banking)* yang mana memerlukan perahasiaan data agar tidak diketahui oleh yang tidak berhak. Pesan yang telah disampaikan dapat dikembalikan lagi ke pesan aslinya hanya oleh orang yang berhak (yang memiliki metode atau kunci untuk mengembalikan isi pesan). Salah satu algoritma penyandian yang dikenal adalah *RSA*, Algoritma *RSA* diperkenalkan oleh Ron Rivest,Adi Shamir dan Len Adleman pada tahun 1976,Teknik dasar penyandian ini sebenarnya sudah ditemukan oleh Clifford Cocks [COCK73] tetapi menjadi rahasia sampai tahun 1977. Algoritma ini menggunakan kunci publik dan kunci pribadi,untuk mengenkripsikan dan mengembalikan data yang telah di enkripsi,Keamanannya berdasarkan tingkat kesusahan untuk memfaktorkan sebuah bilangan bulat yanh sangat besar (*Big Integer)*. Seseorang dapat mencoba memfaktorkan bilangan *RSA* dengan mencoba dengan teknik *Brute Force*,meskipun membutuhkan waktu yang sangat lama,tapi saat ini pemfaktoran bilangan *RSA* telah banyak menunjukkan hasil. Seiring dengan perkembangan kecepatan waktu komputasi pada komputer modern sekarang.

***Kata kunci**: Kriptografi,Brute-Force,keamanan*

1. Pendahuluan

Pesan yang dirahasiakan *(plaintext)* dienkripsi menjadi *chipertext,* yang akan dikembalikan menjadi *plaintext* asal, dalam menyandikan digunakan kunci publik dan kunci private yang tidak sama (algoritma kunci publik) Kunci dari keamanan sebuah penyandian dengan Algoritma RSA terletak pada pemfaktoran bilangan RSA, saat ini belum ditemukan algoritma yang paling magkus untuk memfaktorkan bilangan yang sangat besar belum ditemukan, jika pemfaktoran bilangannya sudah diketahui kita bisa leluasa mengembalikan data yang telah dienkripsi. Pada serangan *brute force* dicoba semua kemungkinan pemfaktoran bilangan tersebut sampai ditemukan. Kenyataannya memfaktorkan bilangan non prima menjadi faktor primanya bukanlah pekerjaan yang mudah. Semakin besar bilangan non primanya tentu semakin sulit pula pemfaktorannya. Semakin sulit pemfaktorannya, Semakin kuat pula algorima RSA.

2. Keamanan RSA

secara ringkas,algoritma RSA adalah sebagai berikut

1. Pilih dua buah bilangan prima sembarang, sebut a dan b ,jaga kerahasiaan a dan b ini
2. Hitung $n = a \times b$, n tidak dirahasiakan.
3. Hitung $m = (a - 1) \times (b - 1)$. Ketika m telah dihitung ,a dan b dapat dihapus.
4 Pilih sebuah bilangan bulat untuk kunci publik,sebut namanya e, yang relatif prima terhadap n (FPB a dan n = 1)
5 Bangkitkan kunci dekripsi, d dengan kekongruenan $ed \equiv 1 \pmod m$.Lakukan enkripsi terhadap isi pesan dengan persamaan $c_i = p_i^{\,e} \bmod n$.dalam hal ini p_i adalah blok plainteks c_i ,adalah chiperteks yang diperoleh,dan e adalah kunci enkripsi (kunci publik).Harus dipenuhi persyaratan bahwa nilai p_i harus terletak dalam himpunan nilai 0, 1, 2, … n -1 untuk menjamin hasil perhitungan tidak berada di luar himpunan

6 Proses dekripsi dilakukan dengan menggunakan persamaan $p_i = c_i^d \bmod n$, yang dalam hal ini d adalah kunci enkripsi.

2.1 Contoh enkripsi RSA

Dalam prakteknya, kita tidak menggunakan bilangan kecil untuk mengekripsikan data,tapi suatu bilangan integer yang sangat besar. Kemudian, daripada mencoba untuk merepresentasikan *plaintext* sebagai integer secara langsung,kita membangkitkan suatu *random session key* dan menggunakannya untuk mengenkripsi *plaintext* secara konvensional, Kita lalu menggunakan kunci publik *session* untuk mengenkripsi hanya session key.
Pengirim pesan A mentransfer pesan ke penerima B dalam format seperti ini :

Kunci enkripsi = xxxx (4 digit)
Plaintext dienkripsi dengan kunci = xxxxxxxxxxxxxxx (12 digit)

Penerima B akan menggunakan *session key* yang telah di enkripsi dan menggunakan kunci *private* nya (n,d) untuk mengembalikan enkripsi.Dia akan menggunakan *session key* dengan algoritma dekripsi simetris untuk mengembalikan enkripsi pesan sebenarnya, pemindahan data ini termasuk dalam detail *plaintext* algoritma yang digunakan.metode *encoding*, *initialisation vectors* dan detil lainnya yang diperlukan oleh si penerima, hanya saja rahasia perlu disimpan seperti kunci private nya.

Misalnya saya berhasil mengintersepsi pemindahan data tersebut,saya bisa mencoba dan melakukan *crack* dari *plaintext* yang di enkripsi secara langsung, atau saya bisa mencoba mengenkripsi atau mendekripsi *session key* dan menggunakannya saat itu, Secara umum sistem ini sekuat ukuran kuat-lemahnya koneksi yang terjadi .

3. Serangan *Brute-Force*

Serangan *brute-force* melawan sebuah *cipher* dengan mencoba seluruh kemungkinan kunci, jika kunci (suatu bilangan) dipilih secar random, *plaintext* akan bisa terlihat setelah sekitar setengah dari seluruh kemugkinan kunci telah dicoba,.

Sejak tahun 2002, *symmetric ciphers* dengan kunci 64 bit atau kurang dari itu bisa diserang *brute-force.*
DES (Data Encription Standard) adalah algoritma penyandian dengan kunci 56bit sudah bisa dipecahkan oleh suatu project yang dinamakan EEF *project* pada awal 1990,mereka bahkan menulis sebuah buku tentang eksploit mereka – *Cracking DES*— O'Reilly dan Assoc. EEF sendiri adalah sebuah lembaga non-profit. Banyak orang merasa bahwa lembaga yang mempunyai sumber dana banyak bisa menyerang dengan *brute-force* sebuah *chipher* simetrik 64 bit.

Dan pasti berhasil . banyak peneliti menyarankan bahwa panjang *key* minimum 128 bit.kunci yang bisa ditembus sekarang paling tidak panjangnya 512bit, untuk kebanyakan algoritma kurva asimetrik,kunci terpanjang yang bisa ditembus lebih pendek,sekitar 128 bit,sebuah pesan yang dienkripsi dengan kunci 109 bit dengan algoritma kurva ellips sudah bisa ditembus dengan serangan *brute-force* pada awal 2003,sampai saat ini panjang kunci 128 bit adalah cukup rasional untuk kurva elips dan 1024 bit untuk algoritma asimetrik seperti RSA,perhatikan bahwa jika panajng kunci 1024 bit adalah 128 karakter dan harus dicoba 2^{1024} kemungkinan kunci.

4. Hasil yang telah dicapai

serangan *brute-force* telah menunjukkan hasil pada pemfaktoran bilangan pada algoritma RSA.bahkan dari pihak RSA nya sendiri telah menjanjikan hadiah sebesar $200.000 atau sekitar 1,9 miliar pada siapa saja yang bisa memfaktorkan bilangan RSA-2048 (617 digit)
http://www.rsasecurity.com/rsalabs/challenges/factoring/challengenumbers.txt

4.1 Pemfaktoran RSA-155

RSA-155 menggunakan 512 bit sebagai kunci dari chiperteks
Pada tanggal 22 agustus 1999,sebuah grup peneliti telah menyelesaikan faktorisasi 155 digit (512 bit) bilangan RSA, pekerjaan ini telah diselesaikan dengan *General Number Field Sieve.* ada sekitar 124 722 179 hubungan dari sebelas *sites* yang berbeda

Dari komputasi yang dilakukan dibangkitkan matriks yang mempunyai 6699191 baris dan 6711336 kolom dan bobot 41732631 (62.27 bilangan bukan nol per baris)

Total waktu dalam pemfaktoran ini sekitar 5,2 bulan (17 maret – 22 agustus) diluar pemilihan waktu polinomial. Jika termasuk pemilihan polinomial maka waktu total nya sekitar 7,1 bulan,dimana RSA 140 membutuhkan sekitar 9 minggu Harus diperhatikan bahwa pemilihan polinomial bisa mengurangi waktu total secara drastis dengan menggunakan lebih banyak lagi mesin.

4.2 Pemfaktoran RSA-576

Pada 3 Desember 2003,sekelompok peneliti di jerman dan beberapa negara lainnya melaporkan kesuksesan memfaktorkan bilangan RSA-576 (174 digit bilangan)

Faktornya adalah:

398075086424064937397125500550386491199064362342526708406385189575946388957261768583317

dan

472772146107435302536223071973048224632914695302097116459852171130520711256363590397527

sebuah metode yang disebut *Lattice sieving* dilakukan oleh J. Franke dan T. Kleinjung menggunakan *Hardware* dari *Scientific Computing Institute* dan *Pure Mathematics.Institute* di kota Bonn dan *Experintal Mathematics Institute* di kota Essen, Jerman.

Line sieving dilakukan oleh P. Montgomery dan H. te Riele di CWI, oleh F. Bahr dan rekan-rekannya, juga oleh NFSNET.*Postprocessing* dilakukan oleh BSI.

RSA-576, sebuah 576-bit atau 174-digit bilangan, adalah bilangan tantangan RSA pertama.bilangan sebelumnya yang berhasil difaktorkan adalan RSA-160 (setelah RSA 155) dari tantangan RSA sebelumnya.,

5 Kesimpulan

Keamanan dari sebuah penyandian RSA terletak pada panjang kuncinya,makin panjang kunci,makin aman, serangan *brute-force* bisa menembus pemfaktoran dari bilangan RSA tersebut,meski dengan kompleksitas waktu dan ruang yang sangat besar, algoritma *brute-force* yang mencoba semua kemungkinan akan mulai menunjukkan hasil ketika sudah setengah atau lebih dari semua kemungkinan dicoba

Dari beberapa data pemfaktoran yang didapat bahwa digunakan banyak mesin yang terhubung dalam satu jaringan juga dalam waktu yang sangat lama (ukuran bulan) ,tetapi waktu ini jelas membantah anggapan bahwa algoritma RSA tidak bisa terpecahkan dalam ukuran bulan.

Daftar Pustaka

[1] C.L Liu ,"*Element of Discrete Mathematics"*,McGraw-Hill,Inc,1985.

[2] Dictionary of Algorithms and data Structures. National Institute of Standards and Technology. http://www.nist.gov/dads/ Diakses tanggal 19 Mei 2005 pukul 13.00 WIB

[3] http://www.cl.cam.ac.uk/users/rnc1/brute.html Diakses tanggal 19 Mei 2005 pukul 13.00 WIB

[4] DES-Project http://www.distributed.net/des/ Diakses tanggal 19 Mei 2005 pukul 13.00 WIB

[5] RSA-*official site Laboratory* http://www.rsasecurity.com/rsalabs/