IMPLEMENTASI ALGORITMA ELGAMAL SEBAGAI PROGRAM *ADD-IN* DI *MICROSOFT WORD*

Agus Hilman Majid

Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
Jalan Ganesha 10 Bandung 40132
e-mail : gushil@gmail.com

ABSTRAK

Pada penelitian kali ini, dilakukan studi terhadap salah satu algoritma kriptografi kunci publik, yaitu algoritma ElGamal. Algoritma ini mendasarkan kekuatannya pada kesulitan memecahkan persoalan matematis logaritma diskrit. Selain itu pada proses transformasi kriptografisnya (enkripsi, dekripsi, tanda tangan digital) digunakan bilangan acak setiap langkah proses sehingga menambah kekuatannya. Selanjutnya dilakukan implementasi dalam bentuk perangkat lunak. Perangkat lunak yang dibangun terdiri dari dua buah, yaitu aplikasi *desktop stand-alone* ElGamal Parameter Generator, dan aplikasi *add-in* komponen *software* Microsoft Word. ElGamal Parameter Generator berfungsi untuk membangkitkan parameter kunci ElGamal, yaitu kunci publik dan kunci privat. Sedangkan aplikasi *add-in* memungkinkan pengguna Microsoft Word untuk memakai fungsi transformasi kriptografis ElGamal langsung dari lingkungan Microsoft Word, melalui toolbar yang disediakan oleh *add-in*, pada dokumen Microsoft Word mereka. Perangkat lunak dikembangkan menggunakan menggunakan tool pengembangan Microsoft Visual Studio 2005 dengan bahasa pemrograman C# dalam lingkungan sistem operasi Microsoft Windows XP Professional. Implementasi menggunakan pustaka eksternal Bouncy Castle Cryptographic C# API untuk menangani masalah aritmatika modulo bilangan yang sangat besar yang digunakan pada algoritma ElGamal, dan *Visual Studio Tools for Office (VSTO)* yang menyediakan *programming interface* terhadap *software* Microsoft Office sehingga memudahkan dalam pengembangannya.Berdasarkan pengujian yang telah dilakukan, perangkat lunak mampu melakukan seluruh fungsinya dengan baik, meliputi proses pembangkitan kunci, enkripsi, dekripsi, pembangkitan tanda tangan digital, pemberian tanda tangan digital terhadap dokumen, dan verifikasi dokumen yang telah ditandatangani secara digital (*digitally signed document*).

Kata kunci: Algoritma Kriptografi ElGamal, *Microsoft Word*, *add-in*, algoritma kriptografi kunci publik, kunci publik, kunci privat, logaritma diskrit, pembangkitan kunci, enkripsi, dekripsi, tanda tangan digital, verifikasi, *Visual Studio Tools for Office (VSTO)*, *desktop*.

Makalah diterima 14 Februari 2007. Revisi akhir 14 Februari 2007.

1. PENDAHULUAN

Pada masa sekarang pertukaran data sangat sering dilakukan, dan mayoritas menggunakan sarana pertukaran yang tidak aman. Sejak dahulu, teknik kriptografi telah dipercaya untuk menangani masalah keamanan data. Oleh karena itu teknik kriptografi merupakan salahsatu teknik yang cocok untuk menangani masalah tersebut.

Dalam makalah ini akan dibahas algoritma kriptografi ElGamal. Algoritma ini termasuk algoritma yang lengkap karena selain dapat digunakan untuk enkripsi dan dekripsi, bisa juga digunakan untuk kepentingan tanda tangan digital. Selanjutnya algoritma ini akan diimplementasikan menjadi sebuah aplikasi *add-in* yang terintegrasi (menjadi bagian) *software Microsoft Word*.

2. ALGORITMA ELGAMAL

Algoritma ElGamal, ditemukan oleh ilmuwan Mesir Taher ElGamal pada tahun 1984, merupakan algoritma kriptografi kunci publik. Algoritma kunci publik menggunakan kunci yang berbeda untuk proses transformasinya. Untuk proses enkripsi menggunakan kunci publik, sedangkan proses dekripsi menggunakan kunci privat. Sebaliknya untuk proses tanda tangan digital menggunakan kunci privat sedangkan untuk verifikasi tanda tangan digital menggunakan kunci publik.Algoritma ElGamal mendasarkan kekuatannya pada fakta matematis kesulitan menghitung logaritma diskrit.

2.1 Logaritma Diskrit

Masalah logaritma diskrit adalah dengan memperhatikan hal berikut ini :

1. Jika diberikan suatu bilangan a, maka menghitung $b \equiv \alpha^a \pmod p$ adalah mudah.
2. Tetapi jika diberikan suatu bilangan b, maka untuk menemukan a sehingga $b \equiv \alpha^a \pmod p$ adalah permasalahan yang sulit.

Selanjutnya bilangan a disebut sebagai logaritma diskrit terhadap b dengan basis α, dinyatakan dengan $a = \log_\alpha b$.

Berikut ini adalah contoh perhitungan logaritma diskrit :

Diberikan p = 17, sehingga F_{17} mempunyai anggota {0,1,2,…,16}. Diberikan generator α = 3, sehingga

F_{17}^* = $\{3^0, 3^1, 3^2, 3^3, 3^4, 3^5, 3^6, 3^7, 3^8, 3^9, 3^{10}, 3^{11}, 3^{12}, 3^{13}, 3^{14}, 3^{15}, 3^{16}\}$

={1, 3, 9, 10, 13, 5, 15, 11, 16, 14, 8, 7, 4, 12, 2, 6,1}

Diberikan nilai a = 10, maka $b \equiv 3^{10} \bmod 17 = 8$

Diberikan nilai b = 14, berapa nilai a sehingga $14 \equiv 3^a \bmod 17$? Dengan melihat himpunan F_{17}^*, kita bisa mendapatkan nilai a = 9.

Hal ini tidak terlalu sulit dilakukan jika p masih merupakan bilangan yang kecil, tetapi merupakan suatu masalah yang sulit jika nilai p adalah nilai yang besar atau bahkan sangat besar. Algoritma kriptografi pada umumnya menggunakan bilangan prima yang sangat besar, sehingga masalah logaritma diskrit pada penggunaannya dalam algoritma kriptografi merupakan suatu permasalahan yang sangat sulit (*infeasible*). Hal ini yang menjadi dasar kekuatan algoritma ElGamal

2.1 Algoritma Kriptografi ElGamal

Berikut ini adalah besaran-besaran yang digunakan pada algoritma ElGamal:

1. p, bilangan prima (tidak rahasia)
2. g, dengan syarat ($g < p$), bilangan acak (tidak rahasia)
3. x, dengan syarat ($x < p$), bilangan acak (rahasia, kunci privat)
4. $y = g^x \bmod p$ (tidak rahasia, kunci publik)
5. m (*plaintext*) (rahasia)

Berikut ini adalah prosedur yang digunakan untuk membangkitkan pasangan kunci publik dan kunci privat:

1. ***Pilih*** sembarang bilangan prima p
2. Pilih dua buah bilangan acak, g dan x, dengan syarat $g < p$ dan $1 \le x \le p-2$
3. Hitung $y = g^x \bmod p$

Hasilnya adalah kunci publik yang berupa tripel (y,g,p) dan kunci privat pasangan (x,p).

Di bawah ini adalah prosedur untuk melakukan enkripsi plaintext menjadi ciperteks:

1. Susun *plaintext* menjadi blok-blok $m_1, m_2, m_3, \ldots$, sedemikian sehingga setiap blok merepresentasikan nilai di dalam selang $[0, p-1]$
2. Pilih bilangan acak k, dengan syarat $1 \le k \le p-2$
3. Setiap blok m_i dienkripsi dengan rumus berikut:

$a_i = g^k \bmod p$

$b_i = y^k m_i \bmod p$

Pasangan a_i dan b_i adalah *ciphertext* untuk blok pesan m_i. Perhatikan, ini menunjukkan bahwa ukuran *ciphertext* yang dihasilkan adalah dua kali ukuran *plaintext*-nya.

Prosedur untuk melakukan dekripsi dari *ciphertext* (a_i,b_i) menjadi blok *plaintext* m_i kembali adalah dengan menggunakan kunci privat x melalui persamaan:

$m_i = b_i / a_i^x \bmod p$

Untuk melakukan proses pemberian tandatangan digital (signing) terhadap pesan M, langkah-langkahnya adalah sebagai berikut:

1. Pilih sembarang angka k, dengan syarat k relatif prima terhadap p – 1
2. Hitung $a = g^k \bmod p$
3. Dengan menggunakan algoritma extended Euclidan, dapat dihitung b dari persamaan M = (xa + kb) mod (p – 1)

Tandatangan digital adalah pasangan (a,b).

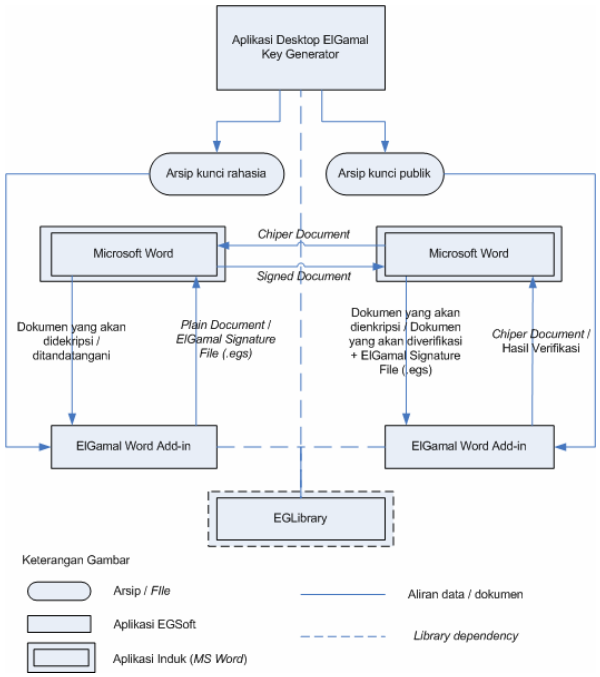
Verifikasi tanda tangan digital dilakukan dengan memeriksa kebenaran persamaan $y^a a^b \bmod p = g^M \bmod p$

3. IMPLEMENTASI

Perangkat lunak, bernama EGSoft, diimplementasikan sebagai tiga bagian yang saling melengkapi, yaitu sebagai berikut :

1. Pustaka ElGamal (EGLibrary), adalah pustaka aplikasi yang berisi operasi-operasi transformasi algoritma ElGamal, yaitu enkripsi, dekripsi, tandatangan digital, dan verifikasi tandatangan digital.
2. Aplikasi *desktop* KeyGenerator (ElGamal Parameter Generator), adalah aplikasi yang bertujuan untuk membangkitkan parameter-parameter ElGamal dan kemudian membentuknya menjadi kunci publik dan kunci privat ElGamal. Kunci-kunci tersebut kemudian akan digunakan oleh aplikasi *MS Word Add-in* untuk melakukan transformasi.
3. Aplikasi *Microsoft Word Add-in*, berupa *add-in* di *Microsoft Word* yang akan melakukan operasi transformasi ElGamal terhadap dokumen tertentu. Aplikasi ini memakai operasi-operasi yang terdapat pada pustaka EGLibrary dan menggunakan kunci publik dan kunci privat hasil pembangkitan aplikasi desktop KeyGenerator sebelumnya.

Berikut ini adalah gambar arsitektur global perangkat lunak EGSoft :



Gambar 1. Layar antarmuka aplikasi MS Word Add-in

Berikut ini adalah rancangan skenario yang terjadi dalam pemakaian perangkat lunak ini:

1. Skenario pembangkitan kunci
   1) Bob melakukan pembangkitan kunci publik dan kunci rahasia / privat ElGamal dengan menggunakan aplikasi ElGamal Parameter Generator.
   2) Bob menyebarkan kunci publik kepada siapa saja yang berkepentingan (umum), termasuk Alice.
   3) Bob menyimpan kunci rahasia dan menjaganya dengan baik.
2. Skenario enkripsi dokumen
   1) Alice akan mengirimkan dokumen yang akan dibaca oleh Bob. Alice menginginkan hanya Bob yang dapat membaca dokumen ini, sehingga Alice berkesimpulan akan mengenkripsi dokumen terlebih dahulu sebelum dikirimkan kepada Bob.
   2) Alice mengenkripsi dokumen dengan ElGamal Word Add-in dan menggunakan kunci publik Bob.
   3) Alice mengirimkan dokumen yang telah terenkripsi kepada Bob.
3. Skenario dekripsi dokumen
   1) Bob mendekripsi dokumen dengan ElGamal Word Add-in dan menggunakan kunci rahasia miliknya.
   2) Bob bisa membaca dokumen asli yang dikirimkan oleh Alice.
4. Skenario tanda tangan dokumen
   1) Bob akan mengirimkan dokumen kepada Alice. Walaupun dokumen yang akan dia kirimkan bukan rahasia (orang lain boleh membacanya), tetapi Bob menginginkan dokumen tersebut sampai ke tangan Alice dengan utuh, tidak terjadi perubahan sama sekali. Bob berkesimpulan untuk menandatangani secara digital terlebih dahulu dokumen yang akan dia kirimkan kepada Alice.
   2) Bob menandatangani dokumen secara digital dengan menggunakan ElGamal Word Add-in dan kunci rahasia miliknya.
   3) Bob mengirimkan dokumen yang telah ditandatangani (dokumen asli + ElGamal Signature File (.egs)) kepada Alice.
5. Skenario verifikasi dokumen yang bertanda tangan
   1) Alice menerima dokumen yang telah ditandatangani (dokumen asli + ElGamal Signature File (.egs)) Bob.
   2) Alice melakukan verifikasi terhadap dokumen tersebut menggunakan ElGamal Word Add-in, *ElGamal Signature File* (.egs) yang dikirimkan bersama dokumen, dan kunci publik Bob untuk memeriksa apakah dokumen yang diterimanya adalah dokumen otentik yang berasal dari Bob dan belum mengalami modifikasi.

Berdasarkan proses analisis dan perancangan, berikut ini adalah kelas-kelas yang diimplementasikan dalam perangkat lunak ini:

1. EGParams (EGLibrary), menangani pembangkitan parameter sistem, shared ***P***,***G***, kunci publik ***Y***, dan kunci privat ***X.***
2. EGProcess (EGLibrary), menangani proses transformasi algoritma ElGamal : enkripsi, dekripsi, tanda tangan digital, verifikasi tanda tangan digital.
3. EGPub (EGLibrary), menyediakan *interface* untuk tipe data kunci publik
4. EGPriv (EGLibrary), menyediakan *interface* untuk tipe data kunci privat
5. EG_Desk (aplikasi desktop), memberikan *interface* bagi pengguna untuk pembangkitan parameter sistem
6. EG_Desk_Gen (aplikasi desktop), menangani hubungan antara parameter kunci yang dibangkitkan oleh EGLibrary dengan aplikasi *desktop*.
7. EG_Word_Conn (*MS Word add-in*), menangani *interface* building pada aplikasi *MS Word* dan menghubungkan *interface* tersebut dengan proses transformasi ElGamal.
8. EG_Word_Proc (*MS Word add-in*), menangani proses transformasi ElGamal terhadap dokumen *Microsoft Word*.

Dalam implementasi EGSoft ini digunakan dua buah *library* dan *tools* eksternal, yaitu:

1. *Bouncy Castle Cryptographic C# API* (http://www.bouncycastle.org). Fungsi-fungsi yang digunakan adalah operasi-operasi dasar seperti operasi-operasi matematika untuk bilangan integer yang sangat besar (BigInteger) dan pembangkitan bilangan acak / *random*.
2. *Visual Studio Tools for Office*. Sekumpulan pustaka yang dibuat di atas *Microsoft .NET* yang memudahkan pengembangan aplikasi berbasis *Microsoft Office*. Dengan menggunakan *tool* ini, pengguna *Visual Studio* dapat membangun aplikasi di atas *MS Office Family* seperti membangung aplikasi *desktop* biasa.

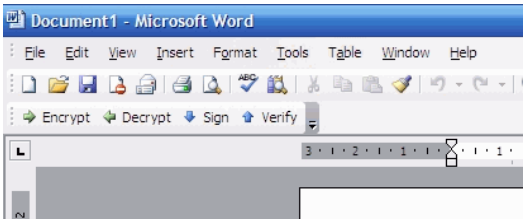
Berikut ini adalah lingkungan perangkat keras tempat implementasi dilakukan:

1. Prosesor Intel Core Duo 1,83 GHz.
2. RAM 512 MB.
3. VGA terintegrasi.

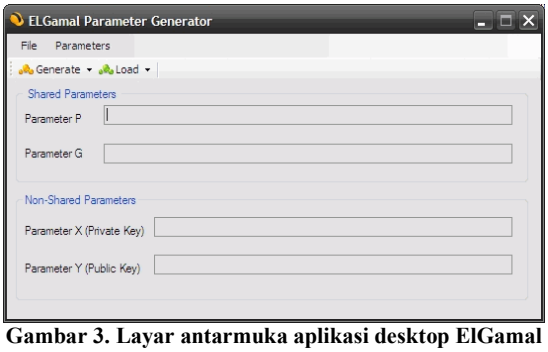
Sedangkan lingkungan perangkat lunaknya adalah sebagai berikut:

1. Sistem Operasi Microsoft Windows XP Professional Edition Service Pack 2.
2. Microsoft Visual Studio 2005, dengan bahasa pemrograman C#.
3. Microsoft Word 2003.

Berikut adalah *screenshot* antarmuka perangkat lunak



Gambar 2. Layar antarmuka aplikasi MS Word Add-in



Gambar 3. Layar antarmuka aplikasi desktop ElGamal Parameter Generator

Penggunaan aplikasi sebagai add-in Microsoft Word akan memberikan kemudahan dan kenyamanan bagi pengguna dalam menggunakan perangkat lunak.

Berdasarkan pengujian yang telah dilakukan dapat disimpulkan bahwa perangkat lunak telah dapat memenuhi fungsi utamanya, yaitu melakukan enkripsi, dekripsi, dan tanda tangan digital pada dokumen Microsoft Word. Demikian juga dapat dilakukan verifikasi terhadap dokumen yang telah ditandatangani.

4. KESIMPULAN

Berikut ini adalah kesimpulan yang dapat diambil:

1. Algoritma ElGamal merupakan algoritma kriptografi yang lengkap, karena selain dapat digunakan untuk proses enkripsi dekripsi, juga bisa dipakai untuk tanda tangan digital dan verifikasinya.
2. Adanya EGLibrary akan mempermudah pengembang yang membutuhkan operasi-operasi algoritma ElGamal dalam perangkat lunaknya.
3. Penggunaan algoritma ElGamal sebagai sarana pengamanan dokumen, terutama dokumen yang dipertukarkan, bisa menjadi suatu alternatif solusi. Hal ini dikarenakan sifatnya sebagai algoritma kunci publik mempunyai kunci berupa pasangan, yaitu kunci publik (yang bisa disebarkan pada publik) dan kunci rahasia (disimpan oleh pemilik dokumen).
4. Implementasi algoritma ElGamal menjadi *Microsoft Word Add-in* memberikan kemudahan bagi pemilik dokumen yang akan mengamankan dokumennya. Pengguna tidak perlu menggunakan aplikasi eksternal, tetapi telah menjadi komponen *MS Word* sehingga penggunaannya tidak memerlukan cara-cara yang sulit.
5. Penggunaan kunci dengan panjang lebih besar memerlukan waktu operasi transformasi yang lebih lama dan hasil transformasi yang lebih besar, sehingga perlu dipertimbangkan panjang kunci yang akan digunakan.

REFERENSI

[1] Munir, Rinaldi, "Bahan Kuliah Kriptografi", Institut Teknologi Bandung, 2004

[2] Schneier, Bruce, "*Applied Cryptography, 2nd Edition*", McGraw-Hill, 1996.

[3] Freedman, Alan, "*Computer Desktop Encyclopedia*", The Computer Languange Co. Inc., 2005.

[4] Menezes, A.J., Van Oorschot, P.C. Vanstone, S.A, "*Handbook of Applied Cryptography*", CRC Press, 1997.

[5] Getz, Ken, "*Understanding the Word Object Model from a Visual Studio 2005 Developer's Perspective*", *<http://msdn2.microsoft.com/en-us/library/aa537164(office.11).aspx>*, Tanggal akses : 20 Desember 2006.

[6] Pressman, Roger S, "*Software Engineering (A Practitional's Approach) 6th Edition*", McGraw-Hill, 2005.

[7] Prayogi, A. Ais, "*Implementasi Identity Based Cryptography Untuk Keamanan Komunikasi Email*", Institutit Teknologi Bandung, 2005.