PERBANDINGAN KRIPTOGRAFI KUNCI PUBLIK MENGGUNAKAN ALGORITMA RSA DAN MATRIKS

Stefanus Astrianto N – NIM : 13504107

Sekolah Tinggi Elektro dan Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if114107@students.if.itb.ac.id

Abstract - *Makalah ini membahas tentang studi terhadap salah satu algoritma kriptografi kunci publik yaitu Teori Matriks, dan membandingkannya dengan algoritma yang sudah sering digunakan oleh umum, yaitu algoritma RSA. Perbandingan yang dibahas adalah mengenai skema autentikasi, ruang kunci, kecepatan, kekuatan dan ekspansi pesan.*

Ide dasar pengembangan teori matriks adalah menggunakan teori pemrogaman linear, yaitu $\mathbf{c} = \mathbf{mG}$*. Dalam hubungan tersebut* $\mathbf{c}$ *adalah katakode (codeword) dari pesan m yang disandikan menggunakan matriks generator* $\mathbf{G}$*. Kaitannya dengan pembuatan disain autentikasi kunci publik, untuk mendapatkan katakode* $\mathbf{c}$*, penelitian ini akan melakukan rekonstruksi terhadap matriks generator* $\mathbf{G}$ *dengan teknik matriks invers.*

Kata kunci*: RSA*, Matriks, kunci publik, kunci privat, enkripsi, dekripsi, kriptografi.

1. Pendahuluan

Masalah keamanan dan keaslian data adalah hal yang penting dalam proses pengiriman data dari seorang pengirim pesan ke penerimanya. Pengirim harus yakin bahwa pesan yang dikirim tersebut utuh, tidak dibaca orang lain, tidak dimodifikasi, dan sampai kepada orang yang tepat. Penerima pesan juga harus yakin bahwa pesan yang diterima tersebut benar, masih asli tanpa modifikasi orang lain.

Atas dasar itulah, ditemukannya sebuah metode penyandian yang melindungi keamanan dan keaslian pesan dalam proses tersebut, yaitu algoritma kunci publik. Proses penandaan suatu dokumen supaya bisa diketahui keasliannya sering disebut sebagai tandatangan digital atau *digital signature*. Hal ini biasa dilakukan untuk mengetahui apakan suatu dokumen pernah dimodifikasi oleh orang lain atau belum.

Intisari dari agoritma kunci publik ini adalah digunakannya 2 macam kunci yang berbeda. Yaitu kunci publik dan kunci privat. Kunci publik adalah kunci yang digunakan untuk mengenkripsi pesan, dan kunci tersebut boleh diketahui oleh orang lain. Kemudian kunci privat adalah kunci yang hanya diketahui oleh penerima pesan saja. Kunci privat digunakan untuk mendekripsikan pesan yang dikirim.

Ibaratnya seperti cara mengirim pesan kepada seorang raja. Raja menyediakan banyak kotak kosong dengan gembok yang masih terbuka. Setiap orang boleh mengambil dan membawa kotak tersebut. Apabila ada yang ingin mengirimkan pesan kepada raja, maka orang tinggal meyimpan surat di dalam kotak tersebut, menutup gemboknya, dan mengembalikannya pada raja. Hanya raja yang bisa membuka kotak tersebut, karena hanya beliau yang mempunyai kuncinya. Kotak tersebut adalah kunci publik, semua orang boleh punya. Tapi hanya yang punya kunci privat saja, yaitu kunci raja yang bisa membukanya.

Ada berbagai macam variasi dari algoritma kunci publik. Shao (1998) telah mengembangkan konsep autentikasi kunci publik berdasarkan faktorisasi dan logaritma diskrit. Jenis kunci publik yang lain adalah RSA yang menggunakan teori bilangan, ElGamal menggunakan logaritma diskrit, Elliptic Curve System yang dikembangkan berdasarkan teori grup, The Merkle-Ellman Knapsack System yang berdasar pada subset sum, dan McEliece public key system yang menggunakan teori koding (Stinson, 1995; Patterson,1987). Sementara itu, matriks invers tergeneralisasi (MIT) juga dapat digunakan untuk mengembangkan sistem kunci publik (Wu and Dawson, 1998).

2. Kunci Publik Menggunakan Teori Matriks

Teori matriks sudah banyak digunakan dalam perhitungan matematis dalam menyelesaikan persamaan-persamaan linier. Salah satu penerapannya adalah pada beberapa bahasa pemrogaman dan penyandian pesan.

Dalam operasinya, metode penyandian dengan matriks menggunakan operasi-operasi antar matriks misalnya penambahan atau perkalian matriks.

Ayres (1982) secara khusus menunjukkan bahwa jika matriks A(*m*,*n*) mempunyai rank *m*, maka

matriks A tersebut mempunyai invers kanan (*right inverse*) matriks Ak(*n,m*), sedemikian hingga AAk = Im, dengan notasi Ak menyatakan invers kanan dari A. Salah satu metode untuk mencari invers kanan dari matriks A tersebut dapat dijelaskan sebagai berikut ini.

1. Pilih submatriks dari A, katakanlah S, yang berdimensi *m*x*m* dan non singular.
2. Cari invers dari S, yaitu S-1.
3. Invers kanan dari A adalah Ak yang berdimensi *n*x*m* di mana : jika matriks S diperoleh dengan jalan menghilangkan kolom ke-*i* dari matriks A, maka baris ke-*i* dari matriks Ak adalah barisnol, sedangkan baris-baris sisanya berasal dari baris-baris S-1 yang bersesuaian dengan kolom-kolom dari matriks A untuk memperoleh matriks S (Ayres Jr, 1982).

Untuk menjamin keamanan data atau pesan m, dari disain **c = mG** dapat dilakukan rekonstruksi terhadap matriks **G**. Model rekonstruksi untuk matriks generator **G** tersebut dapat dijelaskan sbb :

1. Pilih matriks generator **G** = [I_k A], dengan sembarang matriks **A** berdimensi *k*x(*n*-*k*).
2. Sembarang matriks non singular **S** berdimensi *k* x *k*.
3. Pilih sembarang matriks permutasi **P** berdimensi *n* x *n*.
4. **G' = SGP**

Selanjutnya matriks G' ini digunakan untuk melakukan enkripsi pesan m untuk memperoleh ciphertext c, jadi c = mG'. Jadi di sini matriks G' berfungsi sebagai kunci publik untuk melakukan enkripsi pesan m. Sedangkan untuk melakukan dekripsi terhadap ciphertext c untuk mendapatkan pesan m dapat dilakukan komputasi sebagai berikut :

$$\mathbf{c}(\mathbf{P}^{-1}\mathbf{G}^k\,\mathbf{S}^{-1}) = m\mathbf{G}'(\mathbf{P}^{-1}\mathbf{G}^k\,\mathbf{S}^{-1}) = m\mathbf{SGP}(\mathbf{P}^{-1}\mathbf{G}^k\,\mathbf{S}^{-1}) = m.$$

$$R = \mathbf{P}^{-1}\mathbf{G}^k\mathbf{S}^{-1}$$

Jadi di sini matriks $\mathbf{R} = \mathbf{P}^{-1}\mathbf{G}^k\mathbf{S}^{-1}$ berfungsi sebagai kunci pribadi untuk melakukan dekripsi terhadap ciphertext c sehingga mendapatkan pesan m. Jadi, **R** ini harus benar-benar dijaga kerahasiaannya oleh sang penerima pesan.

Apabila ada *intruder* yang ingin mendekripsi pesan tersebut, maka dia harus mengetahui isi dari matriks R. Dimana matriks tersebut misalkan saja berukuran *a* x *b*, sehingga memiliki sejumlah *ab* komponen. Misalnya dalam hal ini kita ambil kasus matriks berisi {0,1}. Maka jumlah kombinasi matriks yang munkin adalah 2^{ab}.. Jadi kekuatan kunci publik oleh matriks berada di jumlah kemungkinan kombinasi pada matriks yang berjumlah sangat banyak, tergantung dimensi dari matriks yang digunakan. Dari hal ini disimpulkan bahwa kunci publik yang dibangkitkan melalui teori matriks cukup sulit untuk dipecahkan, sehingga keamanan dan kerahasiaan pesan dapat dijaga dengan baik.

Sedangkan dari segi kompleksitas enkripsi dan dekripsi data, metode ini mempunyai kompleksitas $O(n^2)$, dan termasuk tingkat kompleksitas yang relatif rendah. Sehingga metode ini dapat digunakan untuk mengenkripsi dan mendekripsi data secara cepat.

3. Algoritma RSA

Salah satu algoritma kunci publik yang paling populer adalah algoritma RSA yang dibuat oleh Ron Rivest, Adi Shamir, dan Leonard Adleman. Keunggulan algoritma ini adalah pada sulitnya memfaktorkan bilangan yang sangat besar menjadi faktor-faktor primanya.

Algoritma RSA mempunyai komponen sebagai berikut :

a.	*p* dan *q* bilangan prima	*rahasia*
b.	$n = p \,.\, q$	*tidak rahasia*
c.	$\phi(n) = (p-1)(q-1)$	*rahasia*
d.	*e* (kunci enkripsi)	*tidak rahasia*
e.	*d* (kunci dekripsi)	*rahasia*
f.	*m* (plainteks)	*rahasia*
g.	*c* (ciphertext)	*tidak rahasia*

Cara pembangkitan kunci pada algoritma RSA :

1. Memilih pasangan bilangan sembarang p dan q.
2. Hitung $n = p \cdot q$. Sebaiknya $p \neq q$, sebab jika $p = q$ maka $n = p^2$ sehingga p dapat diperoleh dengan menarik akar pangkat dua dari n.
3. Hitung $\phi(n) = (p - 1)(q - 1)$.
4. Pilih kunci publik, e, yang relatif prima terhadap $\phi(n)$.
5. Bangkitkan kunci rahasia, yaitu $e.d \equiv 1 \pmod{\phi(n)}$.

Perhatikan bahwa $e.d \equiv 1 \pmod{\phi(n)}$ ekivalen dengan $e \cdot d = 1 + k\phi(n)$, sehingga d dapat dihitung dengan

$$d = \frac{1 + k\phi(n)}{e}$$

k adalah konstanta (1,2,3…) yang menghasilkan nilai d yang bulat.

Dari hasil algoritma di atas, kunci publik yang didapat adalah pasangan (e,n) dan kunci privatnya adalah pasangan (d,n).

Akan terdapat bilangan bulat k yang menyebabkan memberikan bilangan bulat d.

Dalam aplikasi sebenarnya, bilangan-bilangan yang digunakan adalah bilangan yang sangat besar. Sehingga menyulitkan *intruder* untuk mencari pemfaktoran dan menemukan kuncinya. Sebagai efek sampingnya, hal ini kadang membuat kinerja menjadi sedikit melambat. Karena pada prosesnya, bisa terjadi perpangkatan antar bilangan-bilangan besar.

Sebuah RSA dapat disimpulkan aman kalau jumlah n besar. Apabila jumlah n < 256bit, maka kode tersebut akan bisa dipecahkan hanya dalam waktu beberapa jam dengan menggunakan sebuah PC dan program yang bisa didapat secara bebas. Setidaknya n harus berjumlah > 512bit agar kode tersebut baru bisa dipecahkan menggunakan ratusan buah komputer.

Tapi dari segi keamanan, algoritma RSA masih terbukti ampuh melindungi keamanan dan kerahasiaan pesan yang dikirimkan.

```
int rsa_pkcs1_sign( rsa_context *ctx,
                    int mode,
                    int hash_id,
                    int hashlen,
                    unsigned char *hash,
                    unsigned char *sig )
{
    int nb_pad, olen;
    unsigned char *p = sig;

    olen = ctx->len;

    switch( ctx->padding )
    {
        case RSA_PKCS_V15:

            switch( hash_id )
            {
                case RSA_RAW:
                    nb_pad = olen - 3 -
hashlen;
                    break;

                case RSA_MD2:
                case RSA_MD4:
                case RSA_MD5:
                    nb_pad = olen - 3 - 34;
                    break;

                case RSA_SHA1:
                    nb_pad = olen - 3 - 35;
                    break;

                default:
                    return(
XYSSL_ERR_RSA_BAD_INPUT_DATA );
            }

            if( nb_pad < 8 )
                return(
XYSSL_ERR_RSA_BAD_INPUT_DATA );

            *p++ = 0;
            *p++ = RSA_SIGN;
            memset( p, 0xFF, nb_pad );
            p += nb_pad;
            *p++ = 0;
            break;

        default:

            return(
XYSSL_ERR_RSA_INVALID_PADDING );
    }

    switch( hash_id )
    {
        case RSA_RAW:
            memcpy( p, hash, hashlen );
            break;

        case RSA_MD2:
            memcpy( p, ASN1_HASH_MDX, 18 );
            memcpy( p + 18, hash, 16 );
            p[13] = 2; break;

        case RSA_MD4:
            memcpy( p, ASN1_HASH_MDX, 18 );
            memcpy( p + 18, hash, 16 );
            p[13] = 4; break;

        case RSA_MD5:
            memcpy( p, ASN1_HASH_MDX, 18 );
            memcpy( p + 18, hash, 16 );
            p[13] = 5; break;

        case RSA_SHA1:
            memcpy( p, ASN1_HASH_SHA1, 15 );
```

```
            memcpy( p + 15, hash, 20 );
            break;

        default:
            return(
XYSSL_ERR_RSA_BAD_INPUT_DATA );
    }

    return( ( mode == RSA_PUBLIC )
            ? rsa_public(  ctx, sig, sig )
            : rsa_private( ctx, sig, sig )
);
}
```

4. Perbandingan Algoritma RSA dan Matriks

a. Skema Autentikasi

Pada metode matriks, saat ada dua pihak A dan B akan berkirim pesan, setiap pihak harus mempunyai kunci publik masing-masing yang saling ditukarkan, dan kunci privat yang disimpan oleh masing-masing pihak.

	Pihak A	Pihak B
Publik	G'	H
Pribadi	R	F
Enkripsi	$y = mR^T$ $c = yH$	
Dekripsi	-	$y = cF$ $m = (G')^T$

Autentifikasi pada algoritma RSA dilakukan dengan cara menghitung nilai hash dari dokumen yang diinginkan. Kemudian dengan menggunakan kunci publik yang diberikan oleh pengirim, *digital signature* di dekripsi dan dibandingkan dengan hasil fungsi hash dokumen tersebut. Jika hasilnya berubah, maka dokumen tersebut sudah dimodifikasi.

b. Ruang Kunci

Ruang kunci adalah kunci yang dipakai untuk enkripsi maupun untuk dekripsi. Pada metode matriks, kunci-kunci untuk enkripsi adalah **R**(n,k) dari user A dan **H**(n,w) dari user B. Sedangkan kunci-kunci untuk melakukan dekripsi adalah matriks **F**(w,n) dari user B dan **G'**(k,n) dari user A. Jadi total ruang kunci yang diperlukan, baik untuk dekripsi maupun enkripsi adalah n (k + w) bit + n (k + w)bit = 2n (k + w) bit.

Sedangkan pada metode RSA, panjang bit kunci sesuai dengan nilai n yang diberikan. Selama kunci masih relatif prima dengan n. Tetapi panjang kunci yang biasa digunakan > 512bit supaya sulit untuk dipecahkan.

c. Kecepatan

Pada algoritma matriks, sebagian besar waktu digunakan untuk operasi perkalian antar matriks yang biasanya mempunyai komponen sangat banyak, tetapi komponen-komponen tersebut mempunyai ukuran bit yang relatif kecil, sehingga cepat untuk diselesaikan.

Algoritma RSA, dalam operasinya melakukan banyak perpangkatan bilangan-bilangan besar yang memerlukan waktu relatif lebih lama.

d. Kekuatan menghadapi serangan

Inti kekuatan metode matriks terletak pada kemungkinan kombinasi matriks yang sangat besar, yaitu 2^{ab}, dimana ab adalah dimensi matriks. Ditambah lagi dengan variasi matriks permutasi yang sangat luas. Jadi cara memperkuatnya adalah dengan memperbesar dimensi matriks yaitu a dan b.

Inti kekuatan metode RSA terletak pada sulitnya memfaktorkan bilangan yang sangat besar menjadi faktor-faktor primanya. Sehingga cara memperkuatnya adalah dengan memperbesar ukuran bit n.

e. Penggunaan kunci privat dan kunci publik

Pada metode matriks, kunci publik digunakan untuk mengenkripsi pesan dan kunci privat digunakan untuk mendekripsinya.

Sedangkan pada RSA sebaliknya, orang mengenkripsi dengan tujuan memberikan *digital signature*, sehingga menggunakan kunci privat yang hanya dia yang tau, dan semua orang dapat mengetahui siapa pemilik asli dokumen dengan mendekripsi pesan menggunakan kunci publik.

f. Ekspansi pesan

Dalam metode matriks, *palintext* **m** mempunyai panjang *k*, sedangkan *ciphertext* **c** mempunyai panjang *w*. Jadi rasio ekspansi pesan adalah $r = w/k$. Jika nilai *w* cukup besar dibandingkan dengan nilai *k*, maka disain autentikasi ini mempunyai ekspansi pesan yang cukup besar. Dengan kata lain, disain autentiksi ini kurang efisien dalam penggunaan ruang pesan.

Sedangkan pada RSA, setiap proses enkripsi menggunakan fungsi hash, sehingga menghasilkan cipher yang panjangnya relatif sama.

5. Kesimpulan

Dari hasil perbandingan di atas, maka dapat disimpulkan bahwa masing-masing algoritma mempunyai keunggulan dan kelemahan masing-masing.

Algoritma RSA mempunyai keunggulan di sisi autentikasinya yang lebih simpel, serta tidak terlalu banyak memberikan ekspansi pesan. Sedangkan metode matriks mempunyai kelebihan di bidang kecepatan dan ketahanannya dalam menghadapi kemungkinan serangan karena variasi kombinasi matriks yang sangat luas.

6. Daftar Pustaka

Ayre Jr., F., 1982. *Theory and Problems of Matrices*. Asian Edition. Singapore : McGraw-Hill Book Company.

Murtiyasa, B., 2001. Aplikasi Matriks Invers Tergeneralisasi pada Kriptografi *dalam Jurnal Penelitian Sain dan Teknologi* **Vol. 1 Nomor 2** Februari 2001. Hal. 82-90. Surakarta : Lembaga Penelitian UMS.

Shao, Z., 1998. Signature Schemes Based on Factoring and Descrete Logarithms *dalam IEEProceedings Computer Digit. Tech.* **Vol. 145 No. 1**, January 1998, Pp:33-36.

Simmons, G.J.(ed.), 1993. *Contemporary Cryptology : The Science of Information Integrity*, NewYork : IEEE Press.

Stalling, W., 1995. *Network and Internetwork Security Principles and Practice*, New Jersey : Prentice Hall.

Wu, C.K., and Dawson, E., 1998. Generalised Inverses in Public Key Cryptosystem Design *dalam IEE Proceedings Computer Digit. Tech.* **Vol. 145 No. 5**, September 1998, Pp:321-326.