Serangan terhadap Skema Tanda Tangan Digital RSA, ElGamal, Schnorr, dan DSA beserta Teknik untuk Melawan Serangan

Edward Ferdian – NIM : 13503006

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if13006@students.if.itb.ac.id

Abstrak

Digital signature atau tanda tangan digital merupakan sebuah hasil perhitungan kriptografis yang bertujuan untuk menjaga integritas data, otentikasi dan keabsahan sebuah media digital. Metode tanda tangan digital yang akan dibahas dalam makalah ini adalah skema tanda tangan digital RSA, ElGamal, Schnorr, dan DSA. Keempatnya merupakan skema tanda tangan digital. Metode untuk tanda tangan digital ada dua cara, yaitu dengan cara mengenkripsi pesan dan juga dengan cara memberi 'tanda tangan' yang dibangkitkan dengan fungsi *hash* terhadap pesan. Dua algoritma signature yang secara luas digunakan adalah RSA dan ElGamal. RSA menitikberatkan pada rumitnya perhitungan kunci dan sulitnya pemfaktoran kuncinya. Sedangkan pada ElGamal, Schnorr, dan DSA kekuatannya bergantung pada kerumitan dalam hal logaritma diskrit.

Dalam sebuah kriptosistem kunci asimetris, tentunya keamanan sangat diperlukan untuk menjamin keutuhan dan otentikasi data. Berbagai serangan dapat dilakukan terhadap skema tanda tangan digital untuk mengetahui kunci yang digunakan oleh pengirim. Salah satu contoh serangan misalnya Lenstra's Attack yang berbasiskan *Chinese Remainder Theorem*, untuk menyerang RSA. Contoh serangan lainnya adalah penyerangan terhadap skema digital signature berbasis logaritma diskrit seperti ElGamal, Schnorr, dan DSA. Serangan-serangan tersebut tentunya akan menjadi ancaman untuk mengetahui kunci privat pemberi tanda tangan. Dengan memanfaatkan sedikit kesalahan pada tanda tangan yang ada, penyerang dapat mengetahui kunci privat dengan cara melakukan kalkulasi terhadapnya. Selain itu, untuk menghadapi serangan-serangan tersebut, akan dijelaskan pula cara menghadapinya agar algoritma digital signature yang digunakan akan memiliki keamanan dan kerumitan yang cukup sehingga kunci privatnya tetap terjaga.

***Kata kunci**: Digital signature,* ElGamal, RSA, Schnorr, DSA, *Lenstra's Attack*, *discrete logarithm*.

1. Digital Signature

Digital signature atau tanda tangan digital adalah sebuah nilai kriptografis yang bergantung pada pesan dan pada pengirim pesan. Dengan menggunakan tanda tangan digital, maka integritas data dapat dijamin, selain itu tanda tangan digital juga dapat digunakan untuk membuktikan asal pesan, dan nirpenyangkalan.

Hal ini memenuhi beberapa aspek keamanan yang disediakan oleh kriptografi. Adapun empat aspek kemanan tersebut adalah:

1. Kerahasiaan (*confidentiality*) adalah layanan yang digunakan untuk menjaga isi pesan dari siapapun yang tidak berhak untuk membaca pesan tersebut. Dalam kriptografi, biasanya pesan dienkripsi menjadi bentuk yang sulit dimengerti sehingga pesan tetap rahasia.

2. Integritas data (*data integrity*), adalah layanan yang menjamin bahwa pesan masih asli atau belum pernah dimanipulasi selama pengrimina pesan. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi terjadinya manipulasi pesan oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, atau substitusi data lain ke dalam pesan yang sebenarnya.

3. Otentikasi (*authentication*), adalah layanan yang berhubungan dengan identifikasi, baik mengidentifikasi kebenaran pihak-pihak yang berkomunikasi, maupun mengidentifikasi kebenaran sumber pesan. Dua pihak yang saling berkomunikasi juga harus diotentikasi asalnya. Otentikasi sumber pesan secara implisit juga memberikan kepastian integritas data, sebab jika pesan telah dimodifikasi berarti sumber pesan sudah tidak benar. Oleh karena itu, layanan integritas data selalu dikombinasikan dengan layanan otentikasi sumber pesan.

4. Nirpenyangkalan (*non-repudiation*), adalah layanan untuk mencegah entitas yang berkomunikasi melakukan penyangkalan, yaitu pengirim pesan menyangkal melakukan pengiriman atau penerima pesan menyangal telah menerima pesan.

Dari paparan di atas, dapat kita ketahui bahwa tanda tangan digital mampu memenuhi tiga dari empat aspek keamanan di atas, yaitu aspek integritas data, otentikasi, dan nirpenyangkalan. Tanda tangan digital dapat dilakukan dengan dua cara, yaitu:

1. Enkripsi pesan
   Enkripsi pesan dapat diartikan sebagai ukuran otentikasi. Pesan yang telah terenkripsi dapat dikatakan telah ditanda tangani.

2. Tanda tangan digital dengan fungsi *hash*
   Tanda tangan digital dibangkitkan dari *hash* terhadap pesan. Nilai *hash* adalah kode ringkas dari pesan. Tanda tangan digital kemudian ditambahkan pada pesan.

Yang akan dibahas dalam makalah ini adalah, teknik dan algoritma enkripsi untuk nilai *hash* dari pesan, sehinggan menghasilkan tanda tangan digital yang memenuhi tiga kriteria di atas. Selain itu, akan dijelaskan pula beberapa teknik penyerangan yang dapat menyerang tanda tangan digital tersebut.

2. Kriptografi Kunci Publik

Pada kriptografi kunci publik setiap pengguna memiliki sepasang kunci, yaitu kunci publik dan kunci privat. Kunci untuk enkripsi diumumkan kepada publik, digunakan untuk enkripsi dan dilambangkan dengan *e*. Sedangkan kunci untuk dekripsi, bersifat rahasia, disebut kunci privat dan dilambangkan dengan *d*. Karena kunci enkripsi tidak sama dengan kunci dekripsi maka kriptografi kunci publik disebut pula kriptografi asimetri.

Kriptografi kunci asimetri memiliki beberapa keuntungan yaitu :

- Hanya kunci privat yang dijaga kerahasiaannya

- Tidak ada kebutuhan dalam mengirim kunci privat tidak seperti pada sistem kunci simetri

- Pasangan kunci publik / privat tidak perlu diubah dalam periode waktu yang lama
- Dapat digunakan dalam pengiriman kunci simetri
- Beberapa algoritma kunci publik dapat digunakan dalam pemberian tanda tangan digital

Adapun kelemahan yang dimiliki kriptografi kunci asimetri adalah :

- Enkripsi dan dekripsi memakna waktu yang lama karena menggunakan bilangan yang sangat besar
- Ukuran *ciphertext* lebih besar daripada *plaintext*
- Ukuran kunci biasanya lebih besar daripada ukuran kunci simetri
- *Ciphertext* tidak memberikan informasi mengenai otentikasi pengirim

Beberapa algoritma kriptografi kunci publik yang biasa digunakan dalam tanda tangan digital adalah RSA, ElGamal, Schnorr, dan DSA.

2. 1 Algoritma RSA

RSA (Rivest-Shamir-Adleman) merupakan algoritma pertama yang diketahui cocok untuk digunakan dalam proses tanda tangan digital dan juga enkripsi. RSA digunakan secara luas dalam protokol *e-commerce* dan dipercaya masih aman sampai saat ini jika diberikan kunci dengan panjang bit yang cukup besar.

RSA melibatkan dua buah kunci, yaitu kunci publik dan kunci privat. Kunci publik dapat diketahui siapapun dan berguna untuk mengenkripsi pesan. Pesan yang dienkripsi dengan menggunakan kunci publik tersebut hanya bisa didekripsi oleh kunci privatnya. Kunci publik dan privat untuk RSA dapat dibangkitkan dengan cara sebagai berikut:

- Pilih dua buah bilangan prima besar secara acak, p dan q
- Hitung $n = pq$
- Hitung $phi = (p - 1)(q - 1)$
- Pilih bilangan bulat $e > 1$ yang relatif prima terhadap phi
- Hitung d dengan memenuhi persamaan $de \equiv 1 \bmod phi$ atau $de = 1 + k * phi$

Kunci publik adalah pasangan (e, n) sedangkan kunci privat adalah pasangan (d, n).

2. 1. 1 Enkripsi Pesan

Alice mengirim kunci publiknya (n & e) kepada Bob dan merahasiakan kunci privatnya. Bob kemudian akan mengirim pesan M kepada Alice.

Untuk menandatangani pesan M, Bob memecah M menjadi sejumlah $m < n$ lalu ia menghitung *ciphertext* c dengan persamaan :

$$c = m^e \bmod n$$

Setelah menghitung c lalu Bob mengirimkannya kepada Alice.

2. 1. 2 Dekripsi Pesan

Alice dapat mendapatkan pesan m dari c yang dikirim Bob dengan menggunakan kunci privatnya d dengan cara sebagai berikut:

$$m = c^d \bmod n$$

Setelah m diketahui, maka pesan M yang sebenarnya bisa didapatkan.

2. 1. 3 Contoh Implementasi Algoritma RSA

Di bawah ini akan dipaparkan sebuah contoh enkripsi dan dekripsi menggunakan RSA. Untuk memudahkan, pada contoh ini akan digunakan bilangan yang sangat kecil.

- Pilih dua buah bilangan prima $p = 61$ dan $q = 53$
- Hitung $n = pq$, $n = 61 * 53 = 3233$
- Hitung $phi = (p - 1)(q - 1)$, $phi = (61-1)(53-1) = 3120$
- Pilih $e > 1$ yang relatif prima dengan 3120, $e = 17$
- Pilih d yang memenuhi $de \equiv 1 \bmod phi$, $d = 2753$
  $17 + 2753 = 46801 = 1 + 15 * 3120$

Kunci publiknya adalah ($n = 3233$,$e = 17$). Untuk mengenkripsi pesan $m = 123$ maka dapat dlakukan dengan cara:

$$c = m^e \bmod n = 123^{17} \bmod 3233 = 855$$

Kunci privatnya adalah ($n = 3233$,$d = 2753$). Maka fungsi dekripsinya adalah

$$m = c^d \bmod n = 855^{2753} \bmod 3233 = 123$$

Jika Alice menggunakan kunci publik Bob dan mengirimkan pesan yang telah terenkripsi, maka tidak mungkin bagi Bob untuk mengetahui siapa pengirimnya, karena siapapun dapat menggunakan kunci publik Bob.

Keamanan RSA terletak pada dua hal, yaitu masalah pemfaktoran bilangan besar dan masalah RSA. Masalah RSA adalah dalam memecahkan $m^e = c \bmod n$, di mana (e, n) adalah kunci publik RSA dan c adalah *ciphertext*. Cara untuk memecahkan masalah RSA ini adalah memfaktorkan n.

Masalah bilangan besar yang menjadi kekuatan RSA dilakukan dengan memilih bilangan prima besar p dan q. Semakin besar kedua bilangan tersebut, maka akan semakin sulit menyerang algoritma RSA.

Sangatlah penting kunci privat d merupakan bialngan yang sangat besar. Jika p adalah bilangan antara q dan $2q$ dan $d < n^{1/4}/3$, maka d dapat dihitung secara efisien dari n dan e.

Jika dilihat dari masalah kecepatan, RSA jauh lebih lambat dari kriptografi kunci simetris. Selain itu, distribusi kunci sangatlah penting bagi masalah keamanan.

2. 2 Skema ElGamal

Algoritma ElGamal merupakan algoritma enkripsi kunci asimetris untuk kriptografi kunci publik yang didasari kesepakatan kunci Diffie-Hellman. Hal ini diusulkan oleh Taher Elgamal pada tahun 1984. Algoritma ElGamal digunakan pada perangkat lunak *GNU Privacy Guard*, yang merupakan versi dari PGP, dan kriptosistem lainnya. *Digital Signature Algorithm*(DSA) merupakan varian dari skema tanda tangan digital ElGamal, dan tidak sama dengan algoritma ElGamal. ElGamal merupakan skema tanda tangan digital berbasis logaritma diskrit. ElGamal terdiri dari tiga komponen, yaitu pembangkit kunci (*key generator*), algoritma enkripsi, dan algoritma dekripsi.

2. 2. 1 Pembangkitan Kunci

Pembangkit kunci pada ElGamal bekerja dengan cara seperti di bawah ini. Untuk memudahkan, dalam contoh pemberi dan penerima tanda tangan di bawah dilambangkan dengan Alice dan Bob.

- Alice membangkitkan deskripsi dari sekumpulan G dengan urutan q menggunakan pembangkit g.
- Alice memilih nilai x secara acak dari {o, q, ..., q - 1}.

- Alice kemudian menghitung $h = gx$. Alice mem-*publish* h bersama deskripsi dari G, q, g sebagai kunci publiknya, dan mengambil x sebagai kunci privatnya.

2. 2. 2 Algortima Enkripsi

Algoritma enkripsi pada ElGamal bekerja sebagai berikut:

- Untuk mengenkripsi pesan m kepada Alice dengan menggunakan kunci publiknya (G, q, g, h), Bob mengkonversi m menjadi salah satu elemen dari G.

- Bob kemudian memilih nilai y secara acak dari {0, 1, ..., q -1}, lalu menghitung $c_1 = g^y$ dan $c_2 = mh^y$.
- Bob mengirim *ciphertext* (c_1, c_2) kepada Alice

2. 2. 3 Algortima Dekripsi

Untuk mendekripsi *ciphertext* (c_1, c_2) dengan kunci privatnya, yaitu x

- Alice menghitung $c_2(c_1^x)^{-1}$ sebagai pesan *plaintext*

Algortima dekripsi menghasilkan pesan karena

$$c_2(c_1^x)^{-1} = \frac{m \cdot h^y}{g^{xy}} = \frac{m \cdot g^{xy}}{g^{xy}} = m$$

Jika panjang pesan lebih besar dari pada ukuran G, maka pesan tersebut dapat dipecah ke beberapa bagian dan setiap bagian akan dienkripsi masing-masing.

2. 2. 4 Keamanan

ElGamal merupakan contoh sederhana dari sebuah algoritma enkripsi kunci asimetris. Algoritma ini merupakan algoritma yang bergantung pada probabilitas, yang maksudnya adalah sebuah pesan dapat dienkripsi menjadi berbagai kemungkinan *ciphertext,* dengan konsekuensi enkripsi ElGamal biasa menghasilkan 2:1 ekspansi ukuran *plainteks* dan *ciphertext*.

Keamanan ElGamal terletak pada kesulitan dalam memecahkan masalah logaritma diskrit dalam G. Akan tetapi, jika masalah logaritma diskrit ini terpecahkan, maka skema ElGamal pun dapat dipecahkan. Kekuatan ElGamal terletak pada asumsi yang disebut Decisional Diffie-Hellman(DDH). Asumsi ini sering kali lebih kuat daripada asumsi logaritma diskrit

2. 2. 5 Membangkitkan *group* G

Seperti yang telah dijelaskan di atas, ElGamal dapat didefinisikan dari sekumpulan G, dan dapat disebut aman jika asumsi komputasional tertentu (asumsi DDH) bernilai benar. Sayangnya, penggunaan $G = Z_p$ dengan p adalah bilangan prima tidaklah aman, karena asumsi DDH menghasilkan nilai salah untuk kelompok G ini. Menghitung logaritma diskrit sanagt sulit dalam Z_p, tapi hal ini masih belum cukup bagi kemanan ElGamal.

Dua *group* yang paling sering digunakan dalam skema ElGamal adalah *subgroup* dari Z_p dan *group* yang didefinisikan dari kurve *eliptic* tertentu. Berikut ini adalah salah satu contoh cara untuk memilih *subgroup* dari Z_p yang telah dipercaya aman :

- Pilihlah p yang merupakan bilangan prima besar, di mana $p - 1 = kq$, dengan k adalah bilangan *integer* kecil dan q adalah bilangan prima besar. Hal ini dapat dilakukan misalnya dengan $k = 2$, dengan memilih bilangan besar prima q dan memeriksa apakah $p = 2q + 1$ adalah prima.

- Pilih sebuah elemen $g \in Z$ secara acak, dengan $g \neq 1$ dan $gq = 1 \bmod p$ di mana g adalah orde dari q

- Group G adalah subgroup dari Z_p yang dibangkitkan dengan g.

2. 2. 6 Efisiensi

Enkripsi dengan menggunakan skema ElGamal membutuhkan dua eksponensiasi, bagaimanapun kedua eksponensiasi ini bergantung pada pesan dan dapat dihitung terlebih dahulu jika memang diperlukan. Panjang *ciphertext* dua kali panjang plainteks, dan merupakan kekurangan algoritma ini dibandingkan algoritma lainnya. Dekripsi hanya membutuhkan satu eksponensiasi, tidak seperti pada RSA. Dekripsi pada ElGamal tidak dapat dipercepat dengan menggunakan *Chinese Remainder Theorem*.

2. 2. 7 Contoh Implementasi ElGamal

Misalkan Alice membangkitkan pasangan kunci dengan memilih $p = 2357$, $g = 2$, dan x 1751. Alice lalu melakukan perhitungan

$$y = g^x \bmod p = 2^{1751} \bmod 2357 = 1185$$

Sehingga kunci publik Alice adalah ($y = 1185$, $g = 2$, $p = 2357$) dan kunci privatnya adalah ($x = 1751$, $p = 2357$).

Jika Bob ingin mengirim pesan $m = 2035$ kepada Alice, Bob memilih bilangan acak $k = 1520$. Nilai m dan k ada pada selang[0,2357-1] Bob kemudian menghitung

$$a = g^k \bmod p = 2^{1520} \bmod 2357 = 1430$$

$$b = y^k m \bmod p = 1185^{1520}.2035 \bmod 2357 = 697$$

Jadi, *ciphertext* yang dihasilkan adalah (1430, 697).

Jika Alice melakukan dekripsi terhadap pesan tersebut, maka ia melakukan perhitungan

$$(a^x)^{-1} = a^{p-1-x} \bmod p = 1430^{605} \bmod 2357 = 872$$

$$M = b / a^x \bmod p = 697.\ 872 \bmod 2357 = 2035$$

Pesan hasil dekripsi adalah 2035, sama dengan pesan yang dikirim oleh Bob.

2. 3 Skema Tanda Tangan Digital Schnorr

Tanda tangan Schnorr adalah tanda tangan digital yang dihasilkan ileh algoritma tanda tangan Schnorr. Keamanan algoritma ini ada pada sulitnya memecahkan beberapa masalah dalam logaritma diskrit. Skema ini disebut sebagai skema tanda tangan digital yang paling sederhana.

2. 3. 1 Pemilihan parameter

Setiap pengguna skema tanda tangan menyetujui *group G* dengan pembangkit *g* dari bilangan prima *q* di mana masalah logaritma diskritnya sulit dipecahkan. Setiap pengguna lalu menyetujui fungsi *hash* H.

2. 3. 2 Pembangkitan Kunci

- Pilih kunci privat x di mana $0 < x < q$.
- Kunci publik adalah y di mana $y = g^{-x}$

2. 3. 3 Proses Pemberian Tanda Tangan

Untuk menandatangani pesan *m*, dilakukan :

- Pilih bilangan k secara acak dengan memenuhi $0 < k < q$
- Diberi persamaan $r = g^k$
- Diberi persamaan $e = H(M \,||\, r)$
- Diberi persamaan $s = (k + xe) \bmod q$

Tanda tangan digital merupakan pasangan (e, s), dengan $0 \le e < q$ dan $0 \le s < q$, jika *group* Schnorr digunakan dan $q < 2^{160}$, yang berarti tanda tangan ini cukup dalam 40 *byte*.

2. 3. 4 Proses Verifikasi Tanda Tangan

- Diberi persamaan $r_v = g^s y^e$
- Diberi persamaan $e_v = H(M \,||\, r_v)$

Jika $e_v = e$ maka tanda tangan digital bernilai benar. Elemen publiknya adalah G,g,q,y,s,e,r dan elemen privatnya adalah k,x.

2. 4 *Digital Signature Algorithm* (DSA)

Digital Signature Algorithm (DSA) merupakan standar untuk tanda tangan digital. Diusulkan oleh *National Institute of Standards and Technology* (NIST) pada Agustus 1991 untuk digunakan dalam *Digital Signature Standard* (DSS). Berikut ini aka dipaparkan mengenai cara kerja DSA.

2. 4. 1 Pembangkitan Kunci

- Pilih bilangan prima q sepanjang 160 bit
- Pilih bilangan prima p sepanjang L bit, di mana $p = qz + 1$, dengan z adalah *integer* dan $512 \leq L \leq 1024$ dan L dapat dibagi dengan 64.
- Plih h di mana $1<h< p–1$ dan $g = h^2 \bmod p > 1$.
- Pilih x mengunakan metode acak tertentu, di mana $0 < x < q$.
- Hitung $y = g^x \bmod p$
- Kunci publik adalah $\{p, q, g, y\}$, dan kunci privatnya adalah x.

Bila diperlukan $\{p, q, g\}$ dapat digunakan bersama oleh pengguna yang berbeda dalam suatu sistem.

2. 4. 2 Proses Pemberian Tanda Tangan

- Bangkitkan bilangan k secara acak untuk setiap pesan, di mana $0 < k < q$.
- Hitung $r = (g^k \bmod p) \bmod q$
- Hitung $s = (k^{-1}(\text{SHA-1}(m) + x*r)) \bmod q$, di mana SHA-1($m$) merupakan fungsi hash SHA-1 yang diterapkan terhadap pesan m.
- Hitung ulang tanda tangan dengan $r = 0$ *atau* $s = 0$
- Tanda tangan digitalnya adalah (r, s)

2. 4. 3 Proses Verifikasi Tanda Tangan

- Tolak tanda tangan jika $0 < r < q$ atau $0 < s < q$ tidak terpenuhi.
- Hitung $w = (s)^{-1} \bmod q$
- Hitung $u1 = (\text{SHA-1}(m)*w) \bmod q$
- Hitung $u2 = (r*w) \bmod q$
- Hitung $v = ((g^{u1}*y^{u2}) \bmod p) \bmod q$
- Tanda tangan digital dianggap *valid* jika memenuhi $v = r$

Skema DSA ini mirip dengan skema tanda tangan digital ElGamal.

2. 4. 4 Keabsahan Algoritma DSA

Skema tanda tangan digital dikatakan benar jika penguji selalu menerima tanda tangan yang asli. Hal ini dapat ditunjukkan sebagai berkut:

Dari $g = h^2 \bmod p$ dengan mengikuti persamaan $g^q \equiv h^{qz} \equiv h^{p-1} \equiv 1 \pmod p$. Karena $g > 1$ dan q merupakan bilangan prima maka g mengikuti q.

Pemberi tanda tangan melakukan penghitungan

$$S = k^{-1} (\text{SHA-1}(m) + rr) \bmod q$$

di mana

$$k \equiv \text{SHA-1}(m)s^{-1} + xrs^{-1}$$

$$\equiv \text{SHA-1}(m)w + xrw \pmod q$$

Karena g mengikuti q maka didapatkan

$g^k \equiv g^{\text{SHA-1}(m)\,w} g^{x\,r\,w}$

$\equiv g^{\text{SHA-1}(m)\,w} y^{r\,w}$

$\equiv g^{u1} y^{u2} \pmod p$

Pada akhirnya, kebenaran DSA dapat dibuktikan dengan

$R = (g^k \bmod p) \bmod q$

$= g^{u1} y^{u2} \pmod p \bmod q = v$

3. Serangan terhadap Kriptografi Kunci Kunci Publik

3. 1 Penyerangan terhadap Skema RSA

Dalam sebuah RSA, jika $n = pq$, dan p dan q adalah dua buah bilangan prima, e adalah kunci publik dan d adalah kunci privat. Serangan terhadap RSA yang akan dibahas di sini dimisalkan dalam bentuk dekripsi *ciphertext*.

Jika terdapat suatu pesan m, maka *ciphertext* yang didapat dari enkripsi m adalah sebagai berikut :

$$c \equiv m^e \bmod n$$

Jika kita misalkan $d_{t-1}|d_{t-2}|d_{t-3}|d_{t-4}|....d_1|d_0$ adalah representasi bit biner dalam sebuah kunci privat d, dan d_i adalah nilai 1 atau 0 pada bit ke-i, dan $x|y$ melambangkan konkatenasi x dan y, maka :

$$c_i \equiv c^x \bmod n \text{ ,untuk } i = 0, 1, 2,..., t - 1$$

di mana x adalah 2^i.

Jika diketahui c dan d, maka pesan m dapat dituliskan sebagai

$$m \equiv c^d \bmod n$$

atau

$$m \equiv c^d \bmod n \equiv c_{t-1}^{d_{t-1}} \cdots c_i^{d_i} \cdots c_1^{d_1} c_0^{d_0} \bmod n$$

3. 1. 1 Serangan Jenis Pertama

Untuk memudahkan, kita asumsikan dalam mendekripsi sebuah *chipertext*, terdapat kesalahan dalam bit c_i, untuk i yang bernilai acak, $i \in \{0, 1, 2, ..., t - 1\}$, kita lambangkan bit yang mengalami kesalahan tersebut sebagai c'_i. Maka pesan yang akan dihasilkan jika dilakukan dekripsi adalah

$$m' \equiv c_{t-1}^{d_{t-1}} \cdots c_i'^{d_i} \cdots c_1^{d_1} c_0^{d_0} \bmod n$$

Penyerang sekarang telah memiliki m dan m', maka ia dapat menghitung

$$\frac{m'}{m} \equiv \frac{c_i'^{d_i}}{c_i^{d_i}} \bmod n$$

yang bernilai sama dengan (c'_i/c_i) mod n jika $d_i = 1$ atau bernilai sama dengan 1 jika $d_i = 0$. Lalu, kita asumsikan tiap angka yang akan didapat adalah relatif prima terhadap n. Penyerang dapat dengan mudah menghitung semua nilai yang mungkin untuk (c'_i/c_i) mod n, yang berjumlah sebanyak t^2 karena $c'i$ memiliki kemungkinan nilai sebanyak t. Penyerang kemudian membandingkan angka-angka yang didapatnya dengan (m'/m) mod n. Ketika didapatkan sebuah nilai yang sama, maka dapat diketahui i dan d_i adalah 1.

Contoh ini menggambarkan ide dasar dari serangan ini. Contoh ini menunjukkan bahwa kesalahan bit pada lokasi tertentu dapat menyebabkan terbukanya kunci privat.

Contoh di atas mengasumsikan hanya terdapat c_i yang memiliki kesalahan satu bit dalam sebuah *chipertext* dan tidak terdapat kesalahan lain dari c_i sampai c_j, di mana $j > i$.

Sebenarnya jika terjadi kesalahan pada banyak bit dalam *chipertext*, kita tetap dapat menyerang *chiperext* tersebut. Dalam hal ini, dapat dilakukan (m'/m) mod n dengan nilai kemungkinan yang lebih banyak. Misalnya untuk kesalahan pada dua bit, maka (m'/m) mod n dapat disamakan dengan nilai

$$(c'_{i1}\, c'_{i2} / c_{i1}\, c_{i2}) \bmod n$$

di mana $i_1, i_2 \in \{0, 1, 2, ..., t - 1\}$ dan $(c"_i/c'_i)$ mod n di mana $c"_i$ adalah nilai kesalahan dua bit pada c_i. Dalam kasus ini sebanyak $O(t^4)$ kemungkinan nilai dihitung dan disamakan dengan nilai (m'/m) mod n. Secara umum,ada sebanyak t^{2j} kemungkinan nilai yang harus dihitung dalam keadaan kesalahan terdapat pada sejumlah j bit.

3. 1. 2 Serangan Jenis Kedua

Serangan ini dapat dilakukan jika dalam representasi biner dari d terdapat kesalahan dan satu bitnya ditukar(*flip*). Penyerang dapat dengan acak memilih pesan m dan menghitung *chipertext*-nya c. Penyerang kemudian dapat mendekripsi c dan membuat kesalahan bit secara acak(*random bit error*). Diasumsikan bahwa d_i berubah menjadi komplemennya, yaitu d'_i, lalu hasilnya akan menjadi

$$m' \equiv c_{t-1}^{d_{t-1}} \cdots c_i^{d'_i} \cdots c_1^{d_1} c_0^{d_0} \bmod n$$

Karena penyerang memiliki m dan m', maka penyerang tersebut dapat menghitung

$$\frac{m'}{m} \equiv \frac{c_i^{d'_i}}{c_i^{d_i}} \bmod n.$$

Kenyataannya, jika $m'/m \equiv (1/c_i) \bmod n$, maka $d_i = 1$, dan jika $m'/m \equiv c_i \bmod n$, maka $d_i = 0$. Oleh karena itu, penyerang dapat membandingkan m'/m mod n dengan c_i mod n dan $c^{-1}{}_i$ mod n, untuk $i = 0, 1, ..., t - 1$, untuk menentukan satu bit pada d. Penyerang dapat mengulang proses di atas berkali-kali dengan menggunakanan pasangan pesan dan *chipertext* yang sama atau berbeda sampai menghasilkan informasi yang cukup mengenai d.

Jika kesalahan satu bit secara acak terdapat dalam d dalam setiap percobaan, maka secara perhitungan probabilitas, kita bisa mendapatkan kesimpulan, jika kita mengambil percobaan sebanyak t log t, dengan probabilitas lebih besar dari setengah, setiap bit dari d akan terpecahkan.

Perlu diperhatikan jika serangan ini diaplikasikan terhadap *multiple bit errors*(kesalahan pada banyak bit). Kita asumsikan pada kesalahan dua bit. Penyerang perlu membandingkan m'/m mod n dengan c_ic_j mod n, (c_i / c_j) mod n, dan $(1/ c_ic_j)$ mod n , untuk setiap $i, j \in \{0, 1, 2, ..., t - 1\}$. Dalam kasus ini menyamakan m'/m mod n dengan seluruh nilai, memiliki kompleksitas $O(t^2)$ dan bukan $O(t)$ seperti kasus yang memiliki kesalahan pada satu bit.

3. 1. 3 *Lenstra's Attack* pada RSA dengan *Chinese Remainder Theorem*

Lenstra menyerang RSA dengan menggunakan CRA(*Chinese Remainder Algorithm*), dengan memerlukan satu buah tanda tangan digital yang salah dari pesan yang diketahui. Biasanya pada serangan lain, diperlukan dua buah tanda tangan digital, yaitu tanda tangan digital yang benar dan yang salah.

Tanda tangan s dari sebuah pesan m, adalah sama dengan m^d mod n dan s^e mod n adalah sama dengan m mod n. Diketahui bahwa s didapatkan dengan menghitung

$$u \equiv m^d \bmod p$$

dan

$$v \equiv m^d \bmod q$$

dan dengan menggabungkan u dan v menggunakan CRA. Jika kesalahan terjadi saat penghitungan tanda tangan, maka hasilnya, dilambangkan sebagai s' akan tidak memenuhi $m \equiv s'^e \bmod n$. Jika kesalahan hanya terjadi saat komputasi / perhitungan u dan v, dan CRA berhasil dilakukan, maka hasil tanda tangan s' akan memenuhi $s'^e \equiv m \bmod q$, tetapi kongruen yang sama mod p tidak akan memenuhi. Maka, q membagi $s'^e - m$ tapi p tidak membagi $s'^e - m$, sehinggan faktor dari n mungkin ditemukan oleh penerima tanda tangan digital s' yang salah dengan cara menghitung bilangan faktor pembagi terbesar (FPB) dari n dan $s'^e - m$. Serangan ini

sangatlah kuat karena hanya membutuhkan satu buah tanda tangan yang salah tanpa memerlukan tanda tangan yang sebenarnya.

3. 1. 4 Teknik-teknik Penangkal Serangan terhadap RSA

Terdapat beberapa cara dalam menyerang PKC(*Public Key Cryptography*) dengan cara memanfaatkan kesalahan. Cara dalam menyerang ini dapat tergantung kepada algoritma tanda tangan digital yang digunakan atau tidak. Ada dua cara pendekatan yang secara umum digunakan untuk mengatasi serangan. Cara yang pertama berdasar pada prinsip *'check and balance'* dan cara yang kedua berdasar pada prinsip *'information hiding'*.

Cara yang pertama dapat dilakukan dengan memeriksa hasil tanda tangan digital sebelum melakukan pengiriman. Cara yang kedua dapat dilakukan dengan memasukkan algoritma untuk mengacak pada tahap awal komputasi. Berikut ini beberapa metode yang dapat digunakan:

1. Serangan yang dihadapi dapat dihindari dengan melakukan perhitungan terhadap hasil sebanyak dua kali dan mencocokkan hasilnya. Jika hasilnya sama kemungkinan besar hasil tanda tangan digital tersebut memang benar. Akan tetapi pendekatan seperti ini memakan waktu dua kali lipat dari waktu komputasi yang dibutuhkan. Kelemahan dari cara seperti ini adalah bertambahnya waktu komputasi dengan faktor pengali sebesar 2, sehingga banyak yang tidak mau mengimplementasikannya.

2. Dalam banyak kasus, kunci enkripsi e biasanya kecil. Sehingga kita dapat memeriksa hasilnya dengan menghitung $m'^{e} = c \bmod n$. Hal ini dapat diatasi dengan menghitung nilai e yang besar.

3. Dalam beberapa protokol tanda tangan digital, biasanya sebuah *string* yang dipilih secara acak disambungkan ke pesan m yang akan ditanda tangani. Contohnya, jika m adalah sebuah *string* biner dengan panjang 412 bit, maka dipilih bit acak sejumlah 100 bit, yang dilambangkan dengan r, maka hasilnya adalah $(m|r)^{\mathrm{d}} \bmod n$. Karena r bergantung pada panjang pesan dan nilai r berbeda-beda pada tiap kali penghitungan, maka serangan akan sulit memecahkan kunci pada tanda tangan digital.

4. Pada kasus di mana e merupakan bilangan yang sangat besar, sehingga perlu dilakukan perhitungan $c^d \bmod n$ untuk memeriksa keabsahan pesan, maka dapat dilakukan cara ini. Kita dapat memilih bilangan acak sebanyak r digit dan menghitung $r^d \bmod n$. Ini dapat dilakukan kemudian, sebelum c menjadi masukan. Untuk menghitung $c^d \bmod n$, dilakukan terlebih dahulu perhitungan $rc \bmod n$, lalu $rc^d \bmod n$, dan terkahir dilakukan perhitungan $(rc)^d/(r)^d$. Jika tidak terjadi kesalahan, maka hasilnya dapat dipastika benar. Jika terjadi kesalahan pada hasil tersebut disamarkan dengan menggunakan r. Karena r tidak diketahui oleh penyerang dang berbeda utuk setiap dekripsi, penyerangan tidak akan berhasil. Sebagai contoh, pada Serangan Kedua, jika d_i sama dengan 0 atau d'_i sama dengan 1, maka

   $$m'/m \equiv r^x c_i \bmod n.$$

   Karena r tidak diketahui oleh penyerang, maka akan sulit untuk menyerangnya.

Dari metode di atas, dapat disimpulkan bahwa metode 1 sampai 4 dapat melawan berbagai serangan sedangkan hanya metode 1 sampai 3 yang dapat melawan *Lenstra's Attack*.

3. 2 Serangan terhadap Skema berbasis Logaritma Diskrit

Konsep umum serangan terhadap skema RSA dapat diaplikasikan juga untuk menyerang kriptografi kunci publik berbasis logaritma diskrit. Dalam hal ini, kita coba serangan ini pada skema tanda tangan digital ElGamal, skema Schnorr, dan DSA. Dalam subbab ini, kita representasikan kunci privat pemberi tanda tangan sebagai x dan representasi biner dari kunci tersebut sebagai $x_{t-1}|x_{t-2}|...|x_i|...x_1|x_0$ di mana t adalah jumlah bit pada x dan x_i adalah bit ke i pada x. Kunci privat hanya diketahui oleh pemiliknya dan kunci publik dapat diketahui oleh siapapun.

Langkah-langkah yang dilakukan oleh penyerang adalah sebagai berikut:

1. Penyerang melakukan perubahan terhadap bit pada x secara acak untuk mendapatkan tanda tangan yang salah.
2. Penyerang melakukan perhitungan terhadap tanda tangan yang salah untuk mendapatkan sebagian dari kunci privat x.

Penyerang kemudian melakukan langkah 1 dan 2 secara berulang-ulang sampai mendapatkan representasi dari x atau jumlah bit yang cukup besar untuk mencari sisa nilai x secara *brute force*. Untuk menyederhanakan masalah ini, dalam makalah ini ditunjukkan kasus dengan kesalahan satu bit. Selanjutnya akan dibahas pula mengenai kesalahan pada banyak bit.

3. 2. 1 Penyerangan terhadap Skema Tanda Tangan Digital ElGamal

Dalam skema tanda tangan digital ElGamal, untuk menghasilkan pasangan kunci publik dan kunci privat, pertama-tama dipilih bilangan prima p dan dua bilangan acak, g dan x, sehingga g dan x bernilai kurang dari p. Kunci privat yang dihasilkan adalah x dan kunci publiknya adalah ($y \equiv g^z \bmod p, g, p$).

Untuk menghitung tanda tangan digital dari sebuah pesan m, orang yang akan memberikan tanda tangan pertama-tama memilih bilangan k secara acak, di mana k adalah relatif prima terhadap $p - 1$. Selanjutnya dilakukan perhitungan

$$w \equiv g^k \bmod p$$

dan

$$s \equiv (m - xw) / k \bmod (p - 1)$$

Tanda tangan digitalnya merupakan pasangan w dan s. Untuk menyatakan keabsahan tanda tangan digital tersebut, dapat dilakukan perhitungan

$$y^w w^s \equiv g^m \bmod p$$

Diasumsikan bahwa x_i pada x diubah menjadi kebalikannya yaitu x_i pada saat penandatanganan pesan m. Kita lambangkan x sebagai x' karena bit x_i yang salah tersebut. Maka hasilnya akan menjadi

$$w \equiv g^k \bmod p$$

dan

$$s' \equiv (m - x'w) / k \bmod (p - 1)$$

Dengan menggunakan w, s', m, dan kunci publik pemberi tanda tangan (y, p, g), penyerang dapat melakukan perhitungan

$$T \equiv y^w w^{s'} \bmod p \equiv g^m g^{w(x-x')} \bmod p$$

Jika $R_i \equiv g^{wu} \bmod p$ untuk $i = 0, 1, 2, ..., t - 1$, dan untuk $u = 2^i$, maka diperoleh

$$TR_i \equiv g^m \bmod p, \text{ jika } x_i = 0$$

(karena $x_i = 0$, maka $x - x' = -2^i$) dan

$$T / R_i \equiv g^m \bmod p, \text{ jika } x_i = 1$$

(karena $x_i = 0$, maka $x - x' = -2^i$). Penyerang melakukan perhitungan terhadap TR_i dan T / R_i dan memeriksa apakah TR_i atau T / R_i sama dengan $g^m \bmod p$, untuk $i = 0, 1, ..., t - 1$. Jika ditemukan kesamaan, maka satu bit dari x telah ditemukan.

3. 2. 2 Penyerangan terhadap Skema Tanda Tangan Digital Schnorr

Dalam skema tanda tangan digital Schnorr, untuk menghasilkan pasangan kunci publik dan kunci privat, pertama-tama dipilih dua buah bilangan prima p dan q, dengan

$$p = zq + 1$$

untuk nilai q yang sangat besar. Kemudian dipilih sebuah bilangan g yang tidak sama dengan 1, dengan $g^q \equiv 1 \bmod p$. Kunci privat pomberi tanda tangan adalah x yang dipilih secara acak dan kurang dari q, dan kunci publiknya adalah ($y \equiv g^{-x} \bmod p, g, p, q$).
Untuk menghitung tanda tangan digital pada pesan m, maka penanda tangan memilih nilai k secara acak dengan nilai kurang dai q. Setelah itu dilakukan penghitungan

$$w \equiv g^k \bmod p,\ e = h(m|w)$$

dan

$$s \equiv ex + k \bmod q$$

di mana h merupakan fungsi *hash* satu arah yang mengembalikan bilangan kurang dari q. Tanda tangan digital merupakan pasangan e dan s. Karena

$$(g^s y^e \bmod p) = w$$

untuk memverifikasi tanda tangan digital, orang yang melakukan pengujian mendapatkan hasil

$$h(m|(g^s y^e \bmod p)) = e$$

Selama komputasi untuk mendapatkan nilai s, diassumsikan bahwa x_i pada x dibalik menjadi x'_i, dan x yang salah dilambangkan sebagai x'. Maka hasil dari perhitungannya akan menjadi

$$e \equiv h(m|w) \bmod p$$

dan

$$s' \equiv ex' + k \bmod q$$

Dengan menggunakan e, s', m, dan kunci publik pemberi tanda tangan (y, p, g ,q), penyerang melakukan perhitungan

$$T \equiv g^{s'} y^e \equiv wg^{e(x' - x)} \bmod p$$

Jika $R_i \equiv g^{eu} \bmod p$ untuk $i = 0, 1, 2, ..., t - 1$, dan untuk $u = 2^i$, dapat dilihat bahwa

$$TR_i \equiv \mathrm{w}g^{e(z' - z + u)} \bmod p$$

dan

$$T / R_i \equiv \mathrm{w}g^{e(x' - x + u)} \bmod p$$

Maka akan didapatkan
$h(m|(TR_i \bmod p)) = e$, jika $x_1 = 1$

(karena $x_i = 1$, maka $x' - x = -2^i$ dan $TR_i \equiv w \bmod p$) dan

$h(m|(T / R_i \bmod p)) = e$, jika $x_i = 0$

(karena $x_i = 0$, maka $x' - x = 2^i$ dan $T / R_i \equiv w \bmod p$). Maka dengan melakukan iterasi dengan berbagai nilai i dan mencocokkan e dengan $h(m|(TR_i \bmod p))$ dan $h(m|(T / R_i \bmod p))$, penyerang dapat menemukan bit ke i dari kunci privat x.

3. 2. 3 Penyerangan terhadap Skema Tanda Tangan Digital DSA

Dalam skema tanda tangan digital DSA, untuk menghasilkan pasangan kunci publik dan kunci privat, pertama-tama dipilih sebuah bilangan prima p, dengan

$$p = zq + 1$$

untuk nilai prima q yang sangat besar. Kemudian dihitung $g \equiv b^{(p - 1) / q} \bmod p$, di

mana b merupakan bilangan apapun yang kurang dari $p - 1$ dan ($b^{(p-1)/q} \bmod p$) lebih besar dari 1. Kunci privat pemberi tanda tangan adalah x, sebuah bilangan acak yang kurang dari q, dan kunci publiknya adalah ($y \equiv g^{-x} \bmod p, g, p, q$).
Untuk memberi tanda tangan digital pada sebuah pesan m, pemberi tanda tangan memilih terlebih dahulu secara acak sebuah bilangan k yang kurang dari q. Selanjutnya dilakukan perhitungan

$$w \equiv g^{-k} \bmod p \bmod q$$

dan

$$s \equiv (e + wx) / k \bmod q,$$

di mana $e = h(m)$ dengan h merupakan fungsi *hash* satu arah yang menghasilkan sebuah bilangan yang kurang dari q. Tanda tangan digitalnya merupakan pasangan dari w dan s. Untuk memastikan kebenaran tanda tangan, dapat diuji

$$w \equiv g^{(ue \bmod q)} y^{(uw \bmod q)} \bmod p \bmod q$$

di mana $u \equiv 1/s \bmod q$.

Dalam perhitungan s, kita asunsikan bit ke i pada x diubah dari x_i menjadi x'_i . Maka keluaran yang dihasilkan akan menjadi

$$w \equiv g^{-k} \bmod p \bmod q$$

dan

$$s' \equiv (e + wx') / k \bmod q$$

Dengan menggunakan w, $u' \equiv 1 / s' \bmod q$, m, dan kunci publik pemberi tanda tangan (y, p, g, q), penyerang dapat melakukan perhitungan
$e = h(m)$ dan

$$\begin{aligned} T &\equiv g^{(u'e \bmod q)} y^{(u'w \bmod q)} \\ &\equiv g^{(u'(e + xw) \bmod q)} \bmod p \bmod q \end{aligned}$$

Jika diberikan $R_i \equiv g^{(u'wc \bmod q)} \bmod p \bmod q$ untuk $i = 0, 1, 2, ..., t - 1$ dengan c adalah 2^i. Maka didapatkan persamaan

$$TR_i \equiv g^{(u'(e + w(x + c)) \bmod q)} \bmod p \bmod q$$

$$T / R_i \equiv g^{(u'(e + w(z - c)) \bmod q)} \bmod p \bmod q$$

Dari persamaan tersebhut dapat dengan mudah ditunjukkan bahwa

$$TR_i \equiv w \bmod p \bmod q, \text{ jika } x_i = 0$$

$$T / R_i \equiv w \bmod p \bmod q, \text{ jika } x_i = 1$$

Jadi, dengan melakukan iterasi dengan nilai i yang berbeda-beda dan mencocokkan w dengan $TR_i \bmod p \bmod q$ dan $T / R_i \bmod p \bmod q$, penyerang dapat menemukan nilai dari x_i.

3. 2. 4 Kesalahan pada Banyak Bit

Metode-metode penyerangan skema tanda tangan digital berbasis logaritma diskrit di atas dapat dikembangkan untuk kasus dengan banyak kesalahan pada x. Dalam kasus kesalahan satu bit, Rs, dilambangkan sebagai Ri dalam kasus di atas, masing-masing memiliki nilai i masing-masing. Pada kesalahan satu bit komputasinya memiliki kompleksitas $O(2t)$, sedangkan pada kasus kesalahan $j > 1$ pada x, Rs dilambangkan sebagai $R_{i1, i2, ..., ij}$, masing-masing akan memiliki perbedaan sebanyak j dan komputasinya memiliki kompleksitas $O((2t)^j)$.

3. 2. 5 Teknik-teknik Penangkal Serangan Skema Tanda Tangan Digital berbasis Logaritma Diskrit

Teknik penangkal serangan melalui komputasi sebanyak dua kali seperti pada bab sebelumnya juga dapat dilakukan dalam skema ini. Cara lain dalam menangkal serangan pada skema tanda tangan digital ini adalah dengan menyimpan nilai x dan $1 / x$, di mana x digunakan dalam penghitungan s dan $1 / x$ digunakan untuk menguji kebenaran nilai s. Sebagai contoh, kita coba dalam skema Schnorr. Segera setelah menghitung

$$s' \equiv ex' + k \bmod q$$

nilai dari *s'* diverifikasi dengan membandingkan nilai *e* dengan $(s' - k)(1 / x) \bmod q$. Jika kedua nilai ini sama, maka hasilnya dapay dipastikan benar.

Secara umum, untuk mencegah variabel yang salah digunakan dalam perhitungan dan menyebabkian rusaknya sebuah kriptosistem, disarankan variabel tersebut dan inversnya disimpan terlebih dahulu sebelum dilakuka perhitungan. Variable tersebut dapat digunakan dalam perhitungan dan inversnya dapat digunakan untuk memverifikasi hasil perhitungannya.

Sebagai ilustrasi, jika kita ingin memastikan nilai *x* dan *k* yang digunakan pada perhitungan $s = ex + k$, maka kita simpan terlebih dahulu nilai *x*, i / *x*, *k*, 1 / *k* sebelum perhitungan dilakukan. Setelah menghitung $s' = ex' + k'$, dilakukan pengecekan apakah

$$e = k (s' (1 / k) - 1)(1 - x)$$

Nilai *s'* dinyatakan benar jika hanya memenuhi persamaan tersebut.

4. Kesimpulan

Tanda tangan digital berguna dalam menjaga integritas data, otentikasi, dan keabsahan suatu pesan. Penandatanganan suatu media digital, dapat dilakukan dengan berbagai skema. Biasanya proses tanda tangan digital ini menggunakan kriptografi kunci asimetri.

Dua skema tanda tangan digital yang paling banyak digunakan adalah RSA dan ElGamal. Pada RSA, algoritma enkripsi dan dekripsi identik, sedangkan ElGamal merupakan skema yang berbasis logaritma diskrit, sama seperti skema Schnorr dan DSA.

Untuk memberikan tanda tangan digital pada suatu pesan dibutuhkan dua buah kunci, yaitu kunci publik dan kunci privat. Kunci publik digunakan unutuk mengenkripsi pesan, dan kunci privat digunakan untuk mendekripsi pesan. Kunci publik bersifat umum, dan kunci privat bersifat rahasia. Maka dari itu, kunci privat harus dijaga agar tidak disalah gunakan oleh orang lain.

Biasanya penyerang memanfaatkan kesalahan-kesalahan yang ada pada *ciphertext* atau tanda tangan untuk memecahkan kunci privat seseorang. Teknik penyerangan ini berbeda-beda sesuai dengan metode tanda tangan digital yang diserang.

Untuk metode RSA misalnya, penyerang dapat melakukan *Lenstra's Attack* dengan menggunakan *Chinese Remainder Theorem*. Sedangkan untuk skema berbasis logaritma diskrit biasanya serangan didasarkan pada kesalahan bit-bit yang ada pada tanda tangan digital karena gangguan saat proses perhitungan. Dengan berbagai serangan yang dilakukan ini, penyerang dapat mengetahui kunci privat penerima pesan sehingga kunci privat tersebut tidak rahasia lagi dan dapat digunakan oleh penyerang.

Untuk menangkal berbagai serangan tadi, banyak cara yang dapat dilakukan. Pada RSA misalnya, teknik menangkal serangan dapat dilakukan dengan menambah panjang jumlah kunci ataupun dengan melakukan pengecekan terhadap perhitungan tanda tangan digital. Sedangkan pada skema berbasis logaritma diskrit, perlawanan dapat dilakukan dengan cara perhitungan ulang dan verifikasi tanda tangan digital.

5. Daftar Pustaka

1. Munir, Rinaldi. (2004). Bahan Kuliah IF5054 Kriptografi. Departemen Teknik Informatika. Institut Teknologi Bandung.
2. Bao, F., Deng, R. H., Han, Y., Jeng, A. Breaking Public Key Cryptosystems on Tamper Resistant Devices in the Presence of Transient Faults. Institute of Systems Science, National University of Singapore. http://citeseer.ist.psu.edu
3. Anderson, Ross and Needham, Roger. *Robustness Principles for Public Key Protocols*. Cambridge

University Computer Laboratory, Cambridge, England.
4. Zheng, Yuliang and Matsumoto, Tsutomu. *Breaking Smart Card Implementation of ElGamal Signature and Its Variants*. Monash University, Australia; Yokohama National University, Japan.
5. Horster, Patrick; Michels, Markus; Peterson, Holger. 1994. *Efficient Blind Signature Schemes Based on the Discrete Logarith Problem*. Strabe der Nationen, Chemnitz, Germany.
6. Horster, Patrick; Michels, Markus; Peterson, Holger. 1994. *Generalized ElGamal Signatures for One Message Block.* Strabe der Nationen, Chemnitz, Germany.
7. Tsiounis, Yiannis and Yung, Moti. *On the Security of ElGamal Based Encryption*. GTE Laboratories Inc., Waltham, MA; CertCo, New York.
8. ElGamal Encryption http://www.answers.com/topic/elgamal-signature-scheme
9. RSA http://www.answers.com/RSA
10. Discrete logarithm http://www.answers.com/topic/discrete-logarithm
11. Schnorr Signature http://www.answers.com/topic/schnorr-signature
12. Digital Signature Algorithm http://www.answers.com/topic/digital-signature-algorithm