Hyperelliptic Curve Cryptography dan Penggunaannya pada Digital Signature

Alwi Alfiansyah Ramdan – 135 08 099

Program Studi Teknik Informatika
Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
e-mail: alfiansyah.ramdan@gmail.com

ABSTRAK

Makalah ini akan membahas mengenai salah satu pendekatan kriptografi kunci publik dengan menggunakan struktur aljabar dari *hyperelliptic curve* pada *finite fields* untuk melakukan enkripsi dan dekripsi yang disebut dengan *Hyperelliptic Curve Cryptography*. *Hyperelliptic Curve Cryptography* adalah modifikasi dari *Elliptic Curve Cryptography* (ECC) yang menggunakan *elliptic curve* sebagai struktur aljabarnya dimana ECC ini diyakini sebagai salah satu algoritma yang merupakan algoritma kriptografi yang mampu melakukan enkripsi dan dekripsi dengan tingkat keamanan yang tinggi. Makalah juga akan membahas perbandingan *Hyperelliptic Curve Cryptography* dengan *Elliptic Curve Cryptography*. Selain itu akan dibahas pula penggunaan *Hyperelliptic Curve Cryptogarphy* pada *digital signature*.

Kata kunci: *hyperelliptic curve, finite field, Hyperelliptic Curve Cryptography, Elliptic Curve Cryptography, digital signature*.

1. PENDAHULUAN

Pada zaman dahulu, untuk mengirimkan pesan kepada kawan dalam perang sangatlah penting. Keamanan pesan yang dikirim harus dapat terjaga dengan baik agar strategi yang akan diterapkan untuk melawan musuh tidak bocor ke tangan lawan.

Banyak sekali peperangan panjang yang selesai karena pesan yang ditujukan kepada kawan dalam perang tersebut bocor ke tangan lawan sehingga lawan mengetahui apa yang akan dilakukan oleh pihak kawan.

Banyak cara yang dapat dilakukan untuk menjaga keamanan dan keutuhan pesan yang dikirim pada pihak yang berhak menerimanya. Salah satunya adalah dengan menggunakan penyandian, dalam hal ini adalah enkripsi dan dekripsi terhadap pesan yang akan dikirimkan.

Proses enkripsi adalah proses yang mengubah pesan asli menjadi pesan yang tak dimengerti pada saat pengiriman dengan menggunakan sebuah kata kunci. Dan proses dekripsi adalah proses yang mengubah pesan yang tak dimengerti menjadi pesan asli yang bermakna dengan menggunakan kata kunci yang sama pada saat proses enkripsi dilakukan. Hal ini dilakukan agar walaupun pesan jatuh ke pihak lawan, pihak lawan tidak mengerti maksud dari pesan tersebut.

Algoritma untuk melakukan penyandian sudah ada sejak dulu. Ada yang menggunakan besar diameter kayu sebagai kunci untuk melakukan enkripsi dan dekripsi pesan. Pesan tersebut ditulis dalam sebuah pita yang digulung pada kayu tersebut. Untuk dapat membacanya kembali digunakan kayu dengan diameter yang sama.

Semakin berkembangnya peradaban, semakin berkembang pula algoritma yang digunakan untuk melakukan enkripsi dan dekripsi pesan.

Hyperelliptic Curve Cryptography ditemukan pada tahun 1989 oleh Neal Koblitz yang merupakan salah satu penemu Elliptic Curve Cryptosystem pada tahun 1985.

Seperti Erelliptic Curve Crytography, *Hyperelliptic Curve Cryptography* juga dapat dipandang sebagai *Discrete Logatihm Cryptosystem* dimana ruang Z^*_p diganti dengan kumpulan titik dalam sebuah *hyperelliptic curve* pada *finite field*.

Basis matematis keamanan *Hyperelliptic Curve Cryptography* adalah sulitnya melakukan komputasi pada *Hyperelliptic Curve Discrete Logarithm Problem* (HCDLP).

Dalam beberapa kasus, HCDLP akan lebih sulit dipecahkan secara signifikan daripada DLP biasa, namun dalam beberapa kasus, HCDLP ini akan semudah DLP biasa bergantung pada pemilihan *hyperelliptic curve* yang digunakan.

Jika pemilihan *hyperelliptic curve* dilakukan dengan hati-hati, maka kekuatan *Hyperelliptic Curve Cryptography* akan lebih besar daripada *Descrete Logarithm Problem* biasa untuk panjang bit kunci yang sama.

2. *Finite Field* (Bidang Terbatas)

Bidang terbatas (*finite field*) atau yang biasa disebut dengan *Galois Field* (GF) adalah bidang yang hanya memiliki elemen bilangan yang terbatas. Derajat (*order*)

dari *finite field* adalah banyaknya elemen yang ada di dalam *field* (bidang) tersebut. Jika *q* adalah pangkat prima (*prime power*), maka hanya ada satu *finite field* dengan derajat *q*. *Finite field* tersebut dilambangkan dengan F_q atau GF(*q*). Banyak cara dapat digunakan untuk mempresentasikan elemen F_q. Jika $q=p^m$, dimana *p* adalah bilangan prima dan *m* adalah bilangan integer positif, maka *p* disebut dengan karakteristik dari F_q dan *m* disebut sebagai derajat perluasan (*extension degree*) dari F_q.

Pada kebanyakan penggunaan *Hyperelliptic Curve Cryptography*, seperti pada Elliptic Curve Crytography, *finite field* yang digunakan dibatasi pada orde *finite field* berupa bilangan prima (q = bilangan prima), yang dilambangkan dengan F_p atau perpangkatan 2 ($q = 2^m$), dimana pangkat (*m*) adalah integer positif lebih besar daripada satu yang dilambangkan dengan F_2^m.

2.1 *Finite Field* F_p

Bidang terbatas F_p adalah sebuah bidang yang beranggotakan bilangan integer {0,1,2,3,..,*p-1*}, dan *p* merupakan bilangan prima. Setiap perhitungan pada F_p dikalkulasikan dengan modulo *p* agar hasilnya tetap berada dalam daerah F_p. Operasi aritmatika yang beraku pada *finite field* F_p adalah sebagai berikut:

1. Penjumlahan (*Addition*)
   Jika $a,b \in F_p$, maka $a + b = r$, dimana *r* adalah sisa pembagian $a + b$ dengan bilangan prima *p* ($(a + b)\ mod\ p$), sehingga $0 \leq r \leq p-1$. Penjumlahan ini disebut dengan penjumlahan *modulo p* (*mod p*).
2. Perkalian (*Multiplication*)
   Jika $a,b \in F_p$, maka $a * b = s$, dimana *s* adalah sisa pembagian $a * b$ dengan bilangan prima *p* ($(a * b)\ mod\ p$), sehingga $0 \leq s \leq p-1$. Perkalian ini disebut dengan perkalian *modulo p* (*mod p*).
3. Invers (*Inversion*)
   Jika $a,b \in F_p$, maka $a * b = 1$, dimana *b* disebut sebagai *invers modulo a* dari *p*, yang biasa dilambangkan dengan a^{-1}.

Contoh perhitungan:
Finite field F_{23} memiliki elemen {0,1,...,23}. Maka:
$12 + 20 = 9$,
$8 * 9 = 3$,
$8^{-1} = 3$.

2.2 *Finite Field* F_2^m

Bidang terbatas F_2^m biasa disebut dengan *characteristic two Finite Field* atau *binary Finite Field*, dapat dipandang sebagai ruang vektor berdimensi *m* pada F_2. Karena itu, ada himpunan yang beranggotakan *m* elemen $\{\alpha_0, \alpha_1, \alpha_2, .., \alpha_{m-1}\}$ di dalam F_2^m sedemikian sehingga setiap $a \in F_2^m$ dapat ditulis secara unik ke dalam bentuk:

$$a = a_0\alpha_0 + a_1\alpha_1 + \cdots + a_{m-1}\alpha_{m-1}, untuk\ a_i \in \{0,1\} \quad (1)$$

Salah satu cara untuk mempresentasikan elemen-elemen pada F_2^m adalah dengan representasi baris polinomial.

Pada representasi baris polinomial, elemen pada F_2^m merupakan polinomial dengan derajat lebih kecil daripada *m* dengan koefisien bilangan 0 atau 1.

$$\{a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \cdots + a_1x^1 + a_0x^0 |\ a_i : 0,1\} \quad (2)$$

Operasi yang berlaku pada F_2^m dalam representasi basis polinomial adalah sebagai berikut:

1. Penjumlahan (*Addition*)
   Jika $a = (a_{m-1}a_{m-2}...a_1a_0)$ dan $b = (b_{m-1}b_{m-2}...b_1b_0) \in F_2^m$, maka $a + b = c = (c_{m-1}c_{m-2}...c_1c_0)$ dimana $c_i = (a_i + b_i) \bmod 2$. Operasi penjumlahan juga dapat menggunakan operasi XOR untuk setiap elemen *a* dan *b*.
2. Perkalian (*Multiplication*)
   Jika $a = (a_{m-1}a_{m-2}...a_1a_0)$ dan $b = (b_{m-1}b_{m-2}...b_1b_0) \in F_2^m$, maka $a * b = r = (r_{m-1}r_{m-2}...r_1r_0)$ dimana $r_{m-1}x^{m-1} + \cdots + r_1x + r$ adalah sisa dari pembagian $(a_{m-1}x^{m-1} + \cdots + a_1x + a_0) * (b_{m-1}x^{m-1} + \cdots + b_1x + b_0)$ dengan polinomial $f(x)$ pada F_2 (setiap koefisien polinomial direduksi ke modulo 2).

3. *Hyperelliptic Curve*

Sebuah *hyperelliptic curve C* dengan genus *g* pada sebuah finite field F_q dengan karakteristik *p* adalah kurva dengan persamaan bentuk sebagai berikut:

$$y^2 + h(x)y = f(x) \quad (3)$$

dimana $h(x)$ dan $f(x)$ adalah polinomial berkoefisien dalam F_q dan derajat $h(x) \leq g$ dan derajat $f(x) \leq 2g + 1$.

Sebuah syarat ditambahkan, yaitu *C* bukanlah kurva yang singular. Jika $h(x) = 0$ dan $p > 2$, maka $f(x)$ haruslah sebuah polinomial dengan derajat 4. Hal ini menyebabkan, tidak akan ada *x* dan *y* pada F_q yang memenuhi persamaan (3) dan dua persamaan turunan bagian $2y + h(x) = 0$ dan $h'(x)y - f'(x) = 0$ secara bersamaan.

Untuk setiap perluasan *K* dari F_q,

$$C(K) := \{(x,y) \in K \times K \mid y^2 + h(x)y = f(x)\} \cup \{\infty\} \quad (4)$$

Persamaan (4) di atas disebut dengan himpunan titik-titik *K-rational* pada kurva *C*. Titik ∞ disebut dengan *point at infinity* dan titik lainnya disebut dengan *finite points*.

Kelompok objek-objek berbeda yang berhubungan dengan *C*, dimana untuk setiap *field extension K* pada F_q bergabung dalam satu grup disebut dengan *Jacobian of C* (*Jacobian* dari kurva *C*).

Secara singkat, *Jacobian* dari kurva *C* pada *finite field* F_q yang dinotasikan dengan $J_C(F_q)$ adalah sebuah Group Abelian[4] yang akan digunakan dalam perhitungan aritmatik. Penggunaan *Jacobian* pada *Hyperelliptic Curve Cryptography* ini mirip dengan penggunaan grup titik-titik pada *elliptic curve* pada Elliptic Curve Crytography.

Secara formal, *Jacobian* dari kurva *C* adalah sebagai berikut. Untuk sebuah *field K*, dengan $F_q \subset K \subset L$, grup dari titik-titik *K-rational* dari Jacobian pada curva C adalah $Div^0_C(K)/P_C(K)$, dimana, L adalah *closure* aljabar dari F_q, $Div^0_C(K)$ adalah grup *degree-zero-divisor* dari kurva *C* pada *K*, $P_C(K)$ adalah subgrup dari $Div^0_C(K)$, yang merupakan $Div^0_C(K) \cap P_C(L)$.

Seperti pada *elliptic curve*, penting untuk mengetahui seberapa besar grup $J_C(F_q)$. Secara kasar, jumlah elemen pada $J_C(F_q)$ adalah q^g seperti yang dijelaskan pada teorema berikut.

Teorema 1 (Andre Weil). Jika *C* adalah *hyperelliptic curve* pada F_q yang memiliki *genus g*, maka

$$(\sqrt{q} - 1)^{2g} \leq \#J_C(K) \leq (\sqrt{q} + 1)^{2g} \quad (5)$$

4. *Hyperelliptic Curve Cryptography*

Hyperelliptic Curve Cryptography adalah kriptografi kunci publik yang menggunakan persamaan pada *hyperelliptic curve* pada *finite field* untuk melakukan enkripsi dan dekripsi. *Hyperelliptic Curve Cryptography* ini adalah modifikasi dari *Elliptic Curve Cryptography* yang menggunakan *elliptic curve* pada *finite field* sebagai basisnya. Walaupun demikian, HCC tidaklah semudah memperluas / mengganti persamaan kurva pada *elliptic curve* menjadi *hyperelliptic curve*. Grup *Jacobian* yang merupakan basis dimana *Hyperelliptic Curve Cryptography* dibangun jauh lebih kompleks daripada grup titik-titik rasional pada *elliptic curve*.

Karena orde dari grup *Jacobian* dapat dibangun untuk memiliki sebuah faktor prima yang jauh lebih besar pada basis *field* yang kecil, dan ada banyak *hyperelliptic curve* yang cocok untuk aplikasi kriptografi, *Hyperelliptic Curve Cryptography* semakin mendapat perhatian.

Namun demikian, pengguna *Hyperelliptic Curve Cryptography* harus berhati-hati dalam memilih kurva yang tepat. Karena jika salah pilih, ada kemungkinan kriptografi yang digunakan dapat diserang oleh serangan khusus tertentu. Berikut adalah pertimbangan yang harus diperhatikan ketika memilih kurva yang tepat yang akan digunakan untuk kriptografi. Misal *q* adalah bilangan prima maupun $q = 2^l$, dengan *l* adalah bilangan prima. Misalkan *C* adalah *hyperelliptic curve* dengan *genus* 2 pada F_q, dan G adalah grup yang sesuai, dalam hal ini adalah *Jacobian* dari *C* pada F_q ($G = J_C(F_q)$), dan misal *r* adalah pembagi prima terbesar dari jumlah elemen pada G (pembagi prima terbesar dari #G), maka dibutuhkan beberapa hal berikut:

- *r* tidak membagi habis *q*,
- jika $k > 0$ adalah integer terkecil sedemikian sehingga $r|q^k - 1$, maka $k > 20$,
- $r > 2^{160}$,
- $\#G/r \leq 4$

Untuk menghitung #G, harus digunakan algoritma yang efisien. Salah satu algoritma yang efisien adalah algoritma perluasan dari *SEA algorithm* yang dapat menghitung jumlah titik yang sesuai dengan ukuran *field* secara kriptografi. Namun, algoritma inipun masih lambat, sehingga dibutuhkan algoritma tambahan untuk melakukan perhitungan, salah satunya adalah *Complex Multiplication method*, yang akan membangun kurva baru dengan ukuran grup yang diketahui.

Dalam pembuatan sistem kriptografi dengan basis *hyperelliptic curve*, seseorang hanya perlu menemukan satu kurva yang tepat. Sehingga perhitungan jumlah titik hanya perlu dilakukan pada awal pembuatan sistem. Ketika sudah memiliki kurva aman yang sesuai, kurva dapat dipakai selama tidak ada serangan yang mengetahui kurva spesifik yang dipakai. Kurva dan ukuran kurva bukanlah bagian dari kunci kriptografi yang akan dibangun, sehingga meskipun salah satu kunci rahasia terbongkat, kunci rahasia yang lain tidak akan terbongkar.

5. *Hyperelliptic Curve Cryptography Schemes*

Misalkan grup G adalah grup yang elemennya akan digunakan dalam sistem kriptografi yang memiliki permasalahan dalam sulitnya pemecahan *discrete logarithm problem* pada grup tersebut. Dalam hal ini G adalah *Jacobian* dari *hyperelliptic curve* yang digunakan.

5.1 *Key Agreement – Diffie-Hellman Key Agreement Scheme*

Protokol yang digunakan dalam pertukaran kunci rahasia secara aman adalah modifikasi dari sistem pertukaran kunci Diffie-Hellman.

Pada protokol ini, misal A dan B sepakan akan bertukar kunci rahasia dengan komunikasi melalui saluran komunikasi publik yang tak aman. Jangan sampai C yang ingin mengetahui percakapan yang dilakukan A dan B mengetahui pesan yang dikirim antara A dan B.

Pertama-tama, diasumsikan terdapat parameter yang sudah diketahui oleh kedua belah pihak, A dan B, yaitu:

- grup *G*, dalam hal ini adalah $J_C(F_q)$
- sebuah elemen $R \in G$ dengan sebuah orde bilangan prima besar *r*.

Berikut adalah langkah-langkah yang dilakukan A untuk melakukan pertukaran kunci:

1. pilih bilangan integer acak $a \in [1, r-1]$,
2. hitung $P = aR$ yang ada pada grup G, lalu kirim ke B,
3. A menerima elemen $Q \in G$ dari B,
4. hitung $S = aQ$ sebagai kunci rahasia bersama.

Berikut adalah langkah-langkah yang dilakukan A untuk melakukan pertukaran kunci:

1. pilih bilangan integer acak $b \in [1, r-1]$,
2. hitung $Q = bR$ yang ada pada grup G, lalu kirim ke A,
3. B menerima elemen $P \in G$ dari A,
4. hitung $\boldsymbol{S} = bP$ sebagai kunci rahasia bersama.

Dapat dilihat bahwa kedua belah pihak, A dan B telah menghitung nilai *S* yang sama, yaitu sebagai berikut:

$$\boldsymbol{S} = a(bR) = (ab)R = b(aR). \quad (6)$$

Tidak mungkin bagi C untuk dapat menghitng $\boldsymbol{S}$ hanya dengan mengetehui *P*, *Q*, dan *R*, dalam waktu yang capat. Jika C dapat menyelesaikan *discrete logarithm problem* dalam *G*, maka ada kemungkinan C dapat menghitung *a* dari *P* dan *R* dan kemudian dapat menghitung $\boldsymbol{S} = aQ$.

Pasangan (*a*,*P*) disebut sebagai pasangan kunci milik A, dimana *a* adalah kunci privat dan *P* adalah kunci publiknya. Dan pasangan (*b*,*Q*) disebut sebagai pasangan kunci miliki B, dimana *b* adalah kunci privat dan *Q* adalah kunci publiknya.

5.2 *Encryption – The Hyperelliptic Curve Integrated Encryption Scheme*

Skema enkripsi ini menggunakan skema Diffie-Hellman untuk melakukan pertukaran kunci rahasia bersama, dan mengombinasikannya dengan *tools* dari kriptografi kunci simetris untuk lebih memperkuat tingkat keamanan. Hal ini dapat mencegah serangan dengan metode *adaptive choosen ciphertext attacks*.

Pada skema ini, digunakan juga parameter yang digunakan pada skema Diffie-Hellman di atas, yaitu:

- grup *G*, dalam hal ini adalah $J_C(F_q)$
- sebuah elemen $R \in G$ dengan sebuah orde bilangan prima besar *r*.

Tools dari kriptografi kunci publik yang digunakan pada skema ini diantaranya adalah sebagai berikut:

- sebuah fungsi untuk mendapatkan kunci (*key derivation*). Fungsi ini dinotasikan sebagai *KD(P)*, yang mengambil *input* $P \in G$ dan *output* data kunci dengan panjang tertentu,
- sebuah skema kriptografi kunci simetris yang terdiri dari sebuah fungsi enkripsi En_k yang mengenkripsi pesan *M* menjadi cipherteks $C = En_k(M)$ menggunakan kunci *k*, dan fungsi dekripsi De_k yang mendekripsi cipherteks *C* menjadi pesan $M = De_k(C)$,
- sebuah Message Authentication Code MAC_k. Ini bisa dianggap sebagai sebuah fungsi hash berkunci, yang menghitung $MAC_k(C)$ bersesuaian dengan hal berikut. Diberikan pasangan $(C_i, MAC_k(C_i))$, maka dapat dihitung pasangan $(C, MAC_k(C))$ dengan memanfaatkan kunci *k*.

Untuk melakukan enkripsi, A yang ingin mangirimkan pesan rahasia kepada B melakukan langkah-langkah berikut:

1. dapatkan kunci publik Q dari B,
2. pilih angka rahasia secara acak $a \in [1, r-1]$,
3. hitung $C_1 = aR$,
4. hitung $C_2 = aQ$,
5. hitung dua buah kunci k_1 dan k_2 dari $KD(C_2)$, atau $(k_1||k_2) = KD(C_2)$,
6. enkrip pesan menggunakan k_1, $C = En_{k1}(M)$,
7. hitung mac = $MAC_{k2}(C)$,
8. kirim (C_1, C, mac) ke B.

Untuk melakukan dekripsi, B yang menerima pesan rahasia dari A melakukan langkah-langkah berikut:

1. dapatkan pesan (C_1, C, mac) dari A,
2. hitung $C_2 = bC_1$,
3. hiung kunci k_1 dan k_2 dari $KD(C_2)$,
4. cek apakah mac memiliki nilai yang sama dengan $MAC_{k2}(C)$, jika tidak tolak pesan dan stop,
5. dekrip pesan $M = De_{k1}(C)$.

6. *Hyperelliptic Curve Cryptography* dan *Digital Signature*

Penggunaan *Hyperelliptic Curve Cryptography* pada *digital signature* adalah masih jarang digunakan. Biasanya, *Hyperelliptic Curve Cryptography* digabungkan dengan DSA untuk digunakan dalam *digital signature* membentuk algoritma baru bernama *Hyperelliptic Curve Digital Signature Algorithm (HCDSA)*. HCDSA ini adalah varian dari DSA yang menggunakan *Hyperelliptic Curve* sebagai basis perhitungannya.

6.1 *Digital Signature*

Tanda tangan digital dengan menggunakan fungsi *hash* satu arah (*one way hash function*) secara umum memiliki tiga proses utama, yaitu pembangkitan pasangan kunci, pemberian tanda tangan digital (*signing*), dan verifikasi terhadap keabsahan tandatangan digital tersebut (*verifying*).

Pada tahap *signing*, pesan yang hendak dikirim diubah terlebih dahulu menjadi bentuk yang ringkas yang disebut dengan *message diggest* (*MD*) yang diperoleh dengan mentransformasikan pesan *M* menggunakan fungsi *hash* satu arah *H*,

$$MD = H(M). \quad (7)$$

Selanjutnya, MD dienkripsikan dengan algoritma kunci publik menggunakan kunci rahasia (SK) pengirim menjadi tandatangan S,

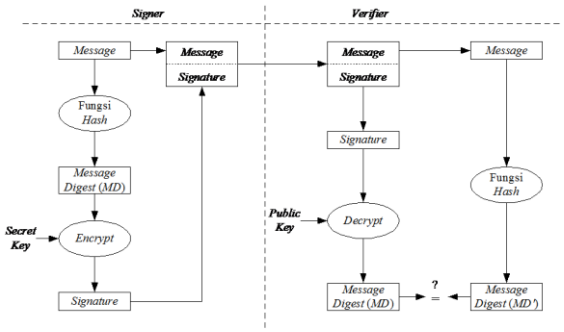
$$S = En_{SK}(MD). \quad (8)$$

Pesan *M* disambung (di-*append*) dengan tanda tangan *S*, lalu keduanya dikirim melalui saluran komunikasi.

Pada tahap *verifying*, pesan *M* dan tanda tangan digital *S* yang dikirim melalui saluran komunikasi telah diterima oleh pihak penerima. Verifikasi dilakukan untuk melakukan otentikasi pesan dengan cara berikut. Tanda tangan *S* didekripsi dengan menggunakan kunci publik (*PK*) pengirim, menghasilkan *message diggest MD*,

$$MD = De_{PK}(S) \quad (9)$$

Penerima kemudian mengubah pesan *M* menjadi *message diggest MD'* menggunakan fungsi *hash* satu arah yang sama dengan yang digunakan pengirim. Jika MD = MD', maka pesan yang diterima otentik dan berasal dari pengirim yang benar.



Gambar 1. Skema *signing* dan *verifying* pada *digital signature*

Proses otentikasi tanda tangan digital ini dapat dijelaskan sebagai berikut.

Apabila pesan M yang diterima sudah berubah, maka MD' yang dihasilkan dari fungsi hash satu arah yang dilakukan penerima akan berbeda dengan MD yang dihasilkan oleh fungsi hash satu arah yang dilakukan pengirim. Jika ini terjadi, maka dapat dikatakan pesan sudah tidak asli lagi (*data integrity*).

Apabila pesan *M* bukan berasal orang yang benar, maka *message diggest MD* yang dihasilkan dari (9) akan berbeda dengan *message diggest MD'* yang dihasilkan saat verifikasi yang terjadi akibat penggunaan kunci publik yang tidak bersesuaian dengan kunci rahasia pengirim pesan. Jika *MD = MD'*, maka pesan yang diterima adalah pesan asli (*message authentication*) dan dikirim oleh orang yang benar (*user authentication*). Karena proses *signing* dan *verifying* menggunakan kunci rahasia dan kunci publik pengirim, maka pengirim tidak bisa menyangkal telah mengirim pesan (*non-repudiation*).

6.2 *Hyperelliptic Curve Cryptography* pada *Digital Signature*

Penggunaan *Hyperelliptic Curve Cryptography* pada *digital signature* ini akan dibagi menjadi 3 langkah skema penggunaan, yaitu skema pembangkitan kunci, skema pemberian tanda tangan digital, dan skema otentikasi.

Pada penggunaan *Hyperelliptic Curve Cryptography* ini, kita akan menggunakan suatu fungsi pemetaan yang memetakan suatu grup yang digunakan dalam perhitungan ke dalam sebuah nilai integer. Dalam hal ini, grup yang digunakan adalah *Jacobian* pada *hyperelliptic curve*. Misal fungsi pemetaan adalah ϕ yang melakukan pemetaan G → **Z**. Misal $D_C = [u(x), v(x)]$ adalah *divisor* dalam bentuk representasi Mumford[5] untuk sebuah *hyperelliptic curve C*. Didefinisikan $u(x) = \sum_{i=0}^{\deg(u(x))} u_i x^i$, $u_i \in F_q$. Maka $\phi(D_C)$ adalah fungsi yang memetakan titik-titik pada *Jacobian* pada *hyperelliptic curve* ke dalam suatu nilai integer.

Diasumsikan terdapat parameter yang sudah diketahui oleh pihak-pihak yang akan melakukan komunikasi melalui pesan rahasia menggunakan tanda tangan digital. Parameter tersebut adalah sebagai berikut.

- grup *G* dalam hal ini adalah $J_C(F_q)$,
- fungsi pemetaan ϕ yang memetakan G → **Z**,
- sebuah elemen $R \in G$ dengan sebuah orde bilangan prima besar *r*,
- sebuah fungsi *hash H*.

6.2.1 Skema Pembangkitan Kunci *Hyperelliptic Curve Cryptography* pada *Digital Signature*

Berikut adalah langkah-langkah yang dilakukan A untuk melakukan pembangkitan sepasang kunci:

1. pilih secara acak sebuah integer $a \in [1, r-1]$,
2. hitung $P = aR$.

Pasangan (*a,P*) disebut sebagai pasangan kunci tanda tangan digital milik A. Dalam hal ini, *a* adalah kunci privat milik A dan *P* adalah kunci publiknya.

6.2.2 Skema Pemberian Tandatangan Digital (*Singning*) *Hyperelliptic Curve Cryptography* pada *Digital Signature*

Berikut adalah langkah-langkah yang dilakukan A untuk memberikan tanda tangan digital pada sebuah pesan rahasia *M*:

1. pilih secara acak sebuah integer $k \in [1, r-1]$,
2. hitung $Q = kR$,
3. hitung *message diggest* $MD = H(M)$,
4. hitung $m = a\,\phi(Q)$,
5. hitung $s \equiv k^{-1}(MD + m) \bmod r$, jika $s = 0$, kembali ke langkah 1.
6. tanda tangan A pada pesan *M* adalah (*M, Q, s*).

(*M, Q, s*) disebut sebagai tanda tangan A pada pesan *M*. Dalam hal ini, *M* adalah pesan yang diberi tanda tangan digital oleh A, *Q* adalah kunci publik tanda tangan digital

milik A, dan *s* adalah tanda tangan digital milik A berkaitan dengan pesan *M*.

6.2.3 Skema Verifikasi Tandatangan Digital (*Verifying*) *Hyperelliptic Curve Cryptography* pada *Digital Signature*

Berikut adalah langkah-langkah yang dilakukan B untuk melakukan verifikasi tanda tangan digital pada sebuah pesan rahasia *M* yang dikirim oleh A:

1. hitung *message diggest MD = H(M)*,
2. hitung $m = \phi(Q)$,
3. hitung $v_1 \equiv s^{-1} MD \bmod r$,
4. hitung $v_2 \equiv s^{-1} m \bmod r$,
5. hitung $V = v_1R + v_2P$,
6. terima tanda tangan jika dan hanya jika $V = Q$.

Untuk melakukan verifikasi terhadap pesan *M* yang dikirim oleh A, diasumsikan B yang ingin melakukan verifikasi tersebut telah memiliki kunci publik tanda tangan *P* milik A.

7. Perbandingan *Hyperelliptic Curve Cryptography* dan *Elliptic Curve Cryptography* dan Analisis

Hyperelliptic Curve Cryptography adalah modifikasi dari *Elliptic Curve Cryptography*. Namun demikian, *Hyperelliptic Curve Cryptography* tidaklah hanya memperluas *Elliptic Curve Cryptography* begitu saja. Berikut adalah perbandingan *Hyperelliptic Curve Cryptography* dengan *Elliptic Curve Cryptography*.

1. *Hyperelliptic Curve Cryptography* menggunakan *hyperelliptic curve* sebagai basisnya. *Elliptic Curve Cryptography* menggunakan *elliptic curve* sebagai basisnya.
2. *Hyperelliptic Curve Cryptography* memiliki persamaan kurva yang lebih rumit daripada *Elliptic Curve Cryptography.*
3. Semua perhitungan pada *Elliptic Curve Cryptography* bergantung pada jenis *finite field* yang digunakan. Perhitungan pada *Hyperelliptic Curve Cryptography* tidak bergantung pada jenis *finite field*, namun lebih bergantung pada nilai karakteristik *field* yang digunakan.
4. Objek perhitungan pada *Hyperelliptic Curve Cryptography* adalah grup *Jacobian* dari kurva yang dipakai. Objek perhitungan pada *Elliptic Curve Cryptography* adalah grup titik-titik rasional pada kurva.
5. Perhitungan pada *Hyperelliptic Curve Cryptography* relatif rumit. Perhitungan pada *Elliptic Curve Cryptography* adalah perhitungan titik-titik rasional biasa.
6. Jika pemilihan kurva tidak tepat, *Hyperelliptic Curve Cryptography* memiliki kemungkinan lebih mudah diserang dari pada *Elliptic Curve Cryptography*.
7. Jika pemilihan kurva tepat, maka *Hyperelliptic Curve Cryptography* jauh lebih sulit ditembus daripada *Elliptic Curve Cryptography*.
8. *Hyperelliptic Curve Cryptography* dan *Elliptic Curve Cryptography* menyebabkan ukuran pesan rahasia yang dikirim manjadi jauh lebih besar daripada ukuran pesan rahasia yang asli.

Sesuai namanya, *Hyperelliptic Curve Cryptography* menggunakan *hyperelliptic curve* sebagai basisnya dan *Elliptic Curve Cryptography* menggunakan *elliptic curve* sebagai basisnya. Hal inilah yang menjadi pembeda utama antara *Hyperelliptic Curve Cryptography* dan *Elliptic Curve Cryptography* yang menyebabkan lebih rumitnya *Hyperelliptic Curve Cryptography*.

Dapat dikatakan, *elliptic curve* merupakan bagian dari *hyperelliptic curve*, karena persamaan *elliptic curve* pada *finite field* F_q, di mana $q = p$ dan p adalah bilangan prima > 3, merupakan persamaan *hyperelliptic curve* dengan $h(x) = 0$, dan $f(x) = x^3 + ax + b$, dan persamaan *elliptic curve* pada *finite field* F_q, di mana $q = 2^m$, merupakan persamaan persamaan *hyperelliptic curve* dengan $h(x) = x$ dan $f(x) = x^3 + ax + b$. Oleh karena itulah ada syarat tambahan untuk *hyperelliptic curve* yaitu jika $h(x) = 0$, maka $f(x)$ haruslah fungsi polinomial dengan derajat minimal 4. Syarat ini ditambahkan agar kurva yang terbentuk bukanlah *elliptic curve* pada *finite field* F_q dengan $q = p$, di mana p adalah bilangan prima > 3.

Dengan perbedaan ini, maka perbedaan dalam semua sistem perhitungan antara *Hyperelliptic Curve Cryptography* dan *Elliptic Curve Cryptography* menjadi sangat jauh berbeda. Salah satunya adalah kebergantungan perhitungan pada *Elliptic Curve Cryptography* terhadap jenis *finite field* yang digunakan. Untuk setiap jenis *finite field* yang digunakan, persamaan kurva serta operasi yang berlaku berbeda satu sama lain. Sedangkan pada *Hyperelliptic Curve Cryptography*, perhitungan tidak bergantung pada jenis *finite field* yang digunakan sehingga persamaan umum kurva tetap sama untuk semua kurva dan operasi yang berlaku adalah operasi umum untuk setiap *hyperelliptic curve*.

Selain itu, dampak juga muncul pada objek perhitungan. Pada *Hyperelliptic Curve Cryptography*, objek perhitungan adalah grup *Jacobian* dari kurva yang dipakai. Sedangkan pada *Elliptic Curve Cryptography*, objek perhitungan adalah titik-titik rasional pada kurva. Inilah yang menyebabkan perhitungan pada *Hyperelliptic Curve Cryptography* menjadi jauh lebih rumit dibandingkan dengan perhitungan pada *Elliptic Curve Cryptography*. Untuk melakukan perhitungan pada suatu

hyperelliptic curve, seseorang harus mendapatkan grup *Jacobian* dari kurva tersebut. Kemudian, dalam melakukan suatu operasi, semua objek operasi adalah anggota atau elemen dari grup *Jacobian* tersebut dengan menggunakan operasi-operasi khusus berkaitan dengan *divisor* kurva yang dipakai, derajat *divisor*, serta grup-grup ataupun subgrup matematika lainnya. Sedangkan untuk melakukan perhitungan pada suatu *elliptic curve*, seseorang hanya perlu menggunakan titik-titik rasional yang ada pada kurva dan melakukan operasi-operasi terhadap titik-titik yang umum, walaupun operasi bergantung pada jenis *finite field* yang digunakan.

Pada *Hyperelliptic Curve Cryptography*, jika pengguna salah dalam memilih kurva yang tepat, maka kemungkinan untuk dapat ditembus lebih besar daripada kesalahan pemilihan kurva pada *Elliptic Curve Cryptography*. Hal ini dimungkinkan terjadi karena perhitungan pada *elliptic curve* bergantung pada jenis *finite field* yang digunakan, sedangkan pada *hyperelliptic curve*, perhitungan tidak bergantung pada jenis *finite field* yang digunakan. Ketika pemilihan kurva tidak tepat, *hyperelliptic curve* yang memiliki persamaan umum yang sama untuk setiap kurva menjadi lebih mudah diserang daripada *elliptic curve* yang persamaan umumnya bergantung kepada jenis *finite field*. Namun demikian, hal ini bergantung pada kurva yang dipilih pada masing-masing algoritma kriptografi di atas.

Sedangkan ketika pemilihan kurva yang digunakan tepat, maka *Hyperelliptic Curve Cryptography* menjadi lebih sulit ditembus daripada *Elliptic Curve Cryptography*. Hal ini dimungkinkan terjadi karena pada *Hyperelliptic Curve Cryptography*, setiap perhitungan melibatkan objek dari grup *Jacobian* dari kurva yang dipakai beserta operasi-operasi rumit lainnya. Sedangkan pada *Elliptic Curve Cryptography*, setiap perhitungannya adalah perhitungan titik-titik rasional kurva pada umumnya yang hanya melibatkan titik-titik rasional yang ada pada kurva yang digunakan. Lagi-lagi, hal ini bergantung pada kurva yang dipilih untuk digunakan sebagai basis pada kriptografi.

Baik *Hyperelliptic Curve Cryptography* maupun *Elliptic Curve Cryptography*, keduanya menghasilkan ukuran pesan rahasia yang membengkak. Hal ini dimungkinkan terjadi karena perhitungan pada kedua algoritma kriptografi menggunakan objek pada suatu grup matematika yang memiliki banyak elemen, grup titik-titik rasional untuk *Elliptic Curve Cryptography* dan grup *Jacobian* untuk *Hyperelliptic Curve Cryptography*. Namun, hal ini mempersulit pihak asing yang ingin menembus kedua algoritma tersebut.

Sampai saat ini, penggunaan *Elliptic Curve Cryptography*, jauh lebih populer dari pada penggunaan *Hyperelliptic Curve Cryptography*. *Elliptic Curve Cryptography* lebih dipilih karena lebih mudah diimplementasikan.

8. Modifikasi *Hyperelliptic Curve Cryptography* dan *Elliptic Curve Cryptography*

8.1 *Latar Belakang*

Seperti yang telah dibahas sebelumnya, *Hyperelliptic Curve Cryptography* menggunakan perhitungan dengan objek dari elemen pada grup *Jacobian* dari sebuah *hyperelliptic curve* yang digunakan beserta operasi-operasi rumit yang berlaku. Hal ini menjadikan *Hyperelliptic Curve Cryptography* sangat rumit dan terkesan merepotkan. Berbeda dengan *Elliptic Curve Cryptography* yang hanya menggunakan titik-titik rasional pada kurva dan operasi titik-titik kurva pada umumnya yang terkesan lebih simpel dan lebih mudah diimplementasikan. Namun demikian, dengan asumsi pemilihan kurva tepat, *Hyperelliptic Curve Cryptography* lebih aman daripada *Elliptic Curve Cryptography*.

Oleh karena itulah, penulis mengusulkan suatu algoritma baru yang merupakan modifikasi dari keduanya. Algoritma ini menggabungkan kompleksitas kurva pada *Hyperelliptic Curve Cryptography* dengan kemudahan operasi pada *Elliptic Curve Cryptography*.

8.2 Modifikasi

Modifikasi *Hyperelliptic Curve Cryptography* dan *Elliptic Curve Cryptography* ini adalah sebagai berikut. Persamaan kurva yang digunakan adalah persamaan *hyperelliptic curve* beserta aturan-aturan yang berlaku padanya. Objek yang digunakan dalam perhitungan adalah titik-titik rasional pada kurva. Operasi-operasi yang digunakan adalah operasi yang berlaku di *elliptic curve* berdasarkan jenis *finite field* yang dipakai. Di sini, hanya akan dibahas penggunaan pada *finite field* F_p. Operasi pada *finite field* F_p telah dijelaskan di atas. Berikut adalah beberapa operasi tambahan yang digunakan.

a. $P + O = O + P = P$ untuk semua P elemen dari kurva C.
b. Jika $P = (x, y)$ elemen dari C, maka $(x, y) + (x, -y) = O$. Titik $(x, -y)$ dinotasikan dengan $-P$, dan disebut negatif dari P.
c. Jika $P = (x_1, y_1)$ elemen dari C dan $Q = (x_2, y_2)$ elemen dari C, dimana $P \neq \pm Q$, maka $P + Q = (x_3, y_3)$, di mana $x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2$ dan $y_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3) - y_1$
d. Jika $P = (x_1, y_1)$ elemen dari C, di mana $P \neq -P$, maka $2P = (x_3, y_3)$, dimana $x_3 = \left(\frac{3x_1^2 + a}{2\,y_1}\right)^2 - 2x_1$ dan $y_3 = \left(\frac{3x_1^2 + a}{2\,y_1}\right)(x_1 - x_3) - y_1$

8.3 Penggunaan Modifikasi pada Proses Enkripsi dan Dekripsi

Penggunaan modifikasi ini pada tanda tangan digital memiliki skema yang sama dengan skema tanda tangan digital pada *Hyperelliptic Curve Cryptography*. Hanya saja grup G yang digunakan adalah kumpulan titik-titik rasional pada kurva dan bukan *Jacobian* dari *hyperelliptical curve* yang digunakan. Sehingga elemen $R \in G$ merupakan sebuah titik.

8.4 Penggunaan Modifikasi pada Digital Signature

Penggunaan modifikasi ini pada tanda tangan digital memiliki skema yang sama dengan skema tanda tangan digital. Beberapa modifikasi diantaranya sebagai berikut

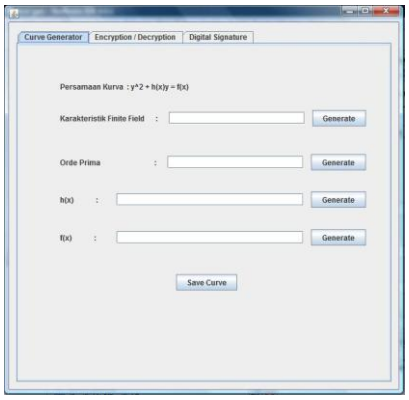
- G dalam hal ini adalah kumpulan titik-titik rasional pada kurva C,
- fungsi pemetaan ϕ yang memetakan suatu titik ke suatu nilai integer (G → **Z**) di mana nilai integer ini adalah nilai titik tersebut pada sumbu x,
- sebuah elemen titik $R \in G$ dengan sebuah orde bilangan prima besar r,
- sebuah fungsi *hash H*.

Dalam hal ini, skema pembentukan sepasang kunci, skema pemberian tanda tangan, dan skema verifikasi sama dengan skema pada *Hyperelliptic Curve Cryptography*.

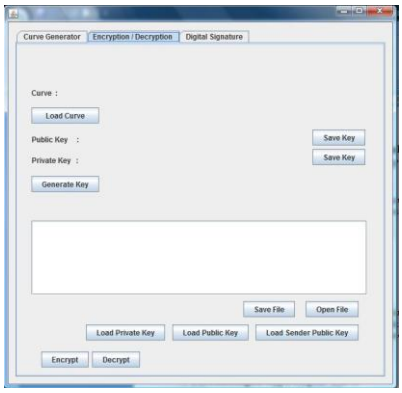
9. Implementasi Modifikasi *Hyperelliptic Curve Cryptography* dan *Elliptic Curve Cryptography*

Penulis mencoba mengimplementasikan modifikasi yang telah disebutkan di atas dalam *platform* Java untuk mengamatinya.

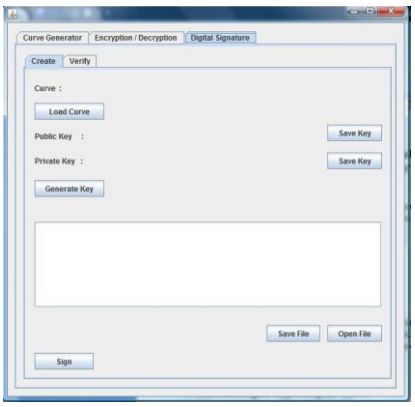
Berikut adalah tampilan program yang telah dibuat.



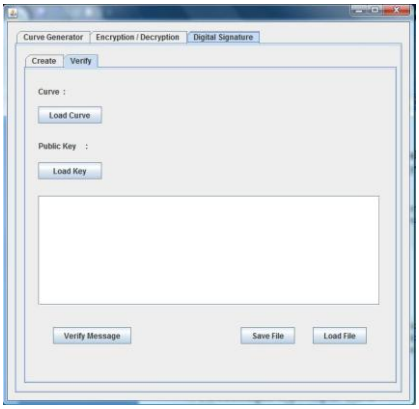
Gambar 2. Tampilan *Curve Generator*



Gambar 3. Tampilan *Encryption/Decryption*



Gambar 4. Tampilan Pemberian Tandatangan



Gambar 5. Tampilan Verifikasi Tandatangan

9.1 Analisis Hasil

Program yang telah dibuat kemudian dijalankan. Ternyata, untuk isi pesan yang panjang, program berjalan dengan lambat dan hasilnya tidak konsisten untuk pesan yang sama baik untuk enkripsi dan dekripsi, maupun untuk penambahan dan verifikasi tandatangan digital.

Untuk isi pesan pendek, program berjalan cepat, namun hasilnya tetap tidak konsisten.

Lambatnya program berjalan kemungkinan karena perhitungan kompleks yang terjadi ditambah panjang pesan setelah dilakukan proses, baik itu enkripsi/dekripsi maupun penambahan dan verifikasi tandatangan digital.

Selain itu, ada kemungkinan algoritma yang dipakai bersifat boros dan tidak mangkus sehingga operasi yang berjalan membebani.

Ketidakkonsistenan yang terjadi dimungkinkan terjadi karena modifikasi tidak meninjau sifat-sifat alami dari *hyperelliptic curve* maupun *elliptic curve* dan langsung menggabungkan keduanya menjadi satu. Penulis juga tidak meninjau sama sekali penggunaan elemen dari grup *Jacobian* pada *hyperelliptic curve* yang digunakan sama sekali yang notabennya adalah objek natural dalam operasi perhitungan pada *hyperelliptic curve*. Ketidakcocokan penggunaan titik-titik rasional pada *hyperelliptic curve* dalam operasi perhitungan yang digunakan juga dapat menjadi sebab ketidakkonsistenan yang terjadi.

10. Serangan pada *Hyperelliptic Curve Cryptography*

Beberapa serangan yang dapat dilakukan untuk menembus *Hyperelliptic Curve Cryptography* adalah sebagai berikut.

1. *Naïve exhaustive search*.
   Dengan metode ini, seseorang melakukan pencarian $R \in G$ secara *brute force* sampai Q atau P didapatkan. Dalam hal ini, metode ini dapat memakan waktu r langkah pada kasus terburuk.
2. *Index Calculus for higher genus curves*.
   Jika *genus g* dari kurva *C* tidak cukup kecil, maka dapat dilakukan index calculus attack pada $J_C(F_q)$.
3. *MOV and Frey-R¨uck Attacks*.
   Misal *r* adalah bilangan prima pembagi terbesar dari sebuah grup *G*, dan *k* adalah integer positif terkecil sedemikian sehingga memenuhi $r|q^k - 1$. Maka dapat ada sebuah *computable injective group homomorphism* dari subgrup *G* orde ke-*r* ke F^*_q.
4. *Anomalous curves*.
   Jika bilangan prima pembagai terbesar *r* dari *#G* sama dengan karakteristik dari F_q, maka seseorang dapat mengubah *dicrete logarithm problem* menjadi *dicrete logarithm problem* dalam grup tambahan dari F_q.
5. Algoritma Pohlig-Hellman.
   Algoritma ini melakukan faktorisasi *r* yang merupakan orde bilangan prima dari *G*, sehingga DLP dalam *G* dapat direduksi menjadi DLP pada subgrup ke-r_i dari *G*.
6. *Weil restriction and cover attacks*.
   Misal *C* adalah *hyperelliptic curve* dengan *genus g* pada finite field F_q^e dan *G* adalah grup $J_C(F_q)$. Maka, terkadang dimungkinkan untuk mencari kurva *X* pada F_q sedemikian sehingga terdapat homomorfism dari *G* ke $J_X(F_q)$ yang mereduksi DLP dari $G = J_C(F_q)$ ke $J_X(F_q)$. Jika *genus* kurva *X* ini tidak lebih besar daripada *e*g*, maka metode *index calculus* dapat dilakukan untuk mencari solusi DLP.
7. Algoritma *baby-step giant-step*
   Algoritma ini memiliki *trade-off* antara waktu dan memori dibandingkan dengan *exhaustive search*. Algoritma ini memerlikan *storage* sebanyak $\sqrt{n}$ titik, dan memerlukan waktu $\sqrt{n}$ pada kasus terburuk.
8. Algoritma Pollard's rho
   Algoritma ini adalah versi acak dari algoritma *baby-step giant-step*. Algoritma ini memerlukan waktu yang hampir sama dengan algoritma *baby-step giant-step* ($\sqrt{\pi n/2}$ langkah), namun memerlukan *storage* yang jauh lebih kecil.

11. KESIMPULAN

Kesimpulan yang dapat diambil antara lain:

1. *Hyperelliptic Curve Cryptography* adalah pendekatan kriptografi kunci publik yang menggunakan *hyperelliptic curve* pada *finite field* sebagai basis dari perhitungannya.
2. *Hyperelliptic Curve Cryptography* merupakan salah satu algoritma kriptografi yang memiliki tingkat keamanan tinggi dengan catatan pemilihan kurva yang digunakan dilakukan secara cermat.
3. *Hyperelliptic Curve Cryptography* pada tanda tangan digital, yang dikenal dengan *Hyperelliptic Curve Digital Signature Algorithm* (HCDSA), masih belum digunakan secara luas karena masih kalah pamor dengan pendahulunya, *Elliptic Curve Cryptography* yang diasosiasikan dengan DSA menjadi ECDSA.

REFERENSI

[1] Hyperelliptic curve, Wikipedia – The Free Encyclopedia, http://en.wikipedia.org/wiki/Hyperelliptic_curve, 2011. Tanggal akses: 26 April 2011, pukul 20.00 WIB.

[2] Hyperelliptic curve cryptography, Wikipedia – The Free Encyclopedia, http://en.wikipedia.org/wiki/Hyperelliptic_curve_ cryptography, 2011. Tanggal akses: 26 April 2011, pukul 20.00 WIB.

[3] Digital Signature Algorithm, Wikipedia – The Free Encyclopedia, http://en.wikipedia.org/wiki/ Digital_Signature_Algorithm, 2011. Tanggal akses: 26 April 2011, pukul 20.00 WIB.

[4] Abelian group, Wikipedia – The Free Encyclopedia, http://en.wikipedia.org/wiki/Abelian_group, 2011. Tanggal akses: 7 Mei 2011, pukul 20.00 WIB.

[5] Rutger Noot. Abelian varieties with *l*-adic Galois representation of Mumford's type. http://www-irma.u-strasbg.fr/~noot/publications/crelle.pdf. Tanggal akses: 7 Mei 2011, pukul 20.00 WIB.

[6] Rinaldi Munir, "Diktat Kuliah IF3058, Kriptografi", Program Studi Teknik Informatika, STEI ITB, 2006.

[7] Elliptic curve, Wikipedia – The Free Encyclopedia, http://en.wikipedia.org/wiki/Elliptic_curve, 2011. Tanggal akses: 7 Mei 2011, pukul 20.00 WIB.

[8] Elliptic curve cryptography, Wikipedia – The Free Encyclopedia, http://en.wikipedia.org/wiki/Elliptic_curve_cryptography, 2011. Tanggal akses: 7 Mei 2011, pukul 20.00 WIB.

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, 13 Maret 2011

Alwi Alfiansyah Ramdan
135 08 099