Analisis Penggunaan *Elliptic Curve Cryptography* pada *Digital Signature*

Aulia Rahma Amin
13503009
Email : aulia@students.itb.ac.id
Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung

Abstrak

Elliptic Curve Digital Signature Algorithm (ECDSA) adalah *Elliptic Curve* yang analog dengan *Digital Signature Algorithm* (DSA). DSA ditetapkan sebagai standard ANSI pada tahun 1999 dan ditetapkan sebagai standard IEEE dan NIST pada tahun 2000. Dan pada tahun 1998 DSA diterima sebagai standard ISO. Tidak seperti permasalahan logaritma diskrit dan pemfaktoran bilangan bulat biasa, permasalahan logaritma diskrit pada *Elliptic Curve* tidak mengenal adanya perpangkatan waktu secara sub-eksponensial untuk memecahkannya. Oleh karena itu, kekuatan algoritma *Elliptic Curve* lebih besar untuk panjang bit kunci yang sama. Makalah ini menjelaskan mengenai *Elliptic Curve Digital Signature Algorithm* (ECDSA) dan mendiskusikan implementasinya.

Kata Kunci : *Elliptic Curve Cryptography, Digital Signature*

1. Pendahuluan

Digital Signature Algorithm (DSA) diperkenalkan dalam *Federal Information Processing Standard* (FIPS) oleh pemerintah Amerika Serikat, yang dikenal dengan *Digital Signature Standard* (DSS). Keamannya bergantung pada kesulitan dalam memecahkan permasalahan logaritma diskrit (*Discrete Logarithm Problem*) pada bilangan prima dalam ruang Z_p^*.

Elliptic Curve Cryptosystem (ECC) ditemukan oleh Neal Koblitz dan Victor Miller pada tahun 1985. ECC dapat dipandang sebagai *Discrete Logarithm* (DL) *cryptosystem* dimana ruang Z_p^* diganti dengan kumpulan titik pada sebuah kurva elips (*Elliptic Curve*) pada bidang terbatas. Basis matematis keamanan *Elliptic Curve cryptosystem* adalah kesulitan dalam komputasi *Elliptic Curve Discrete Logarithm Problem* (ECDLP).

Karena ECDLP secara signifikan lebih sulit dipecahkan daripada DLP, maka kekuatan sistem *Elliptic Curve* jauh lebih besar daripada sistem logaritma diskrit konvensional untuk panjang bit kunci yang sama. Oleh karena itu, dengan parameter yang lebih kecil namun dengan tingkat keamanan yang sama dapat diperoleh apabila kita menggunakan ECC dibanding dengan menggunakan sistem DL konvensional. Keuntungan yang bisa didapatkan dengan parameter yang lebih kecil adalah kecepatan komputasi lebih tinggi, penggunaan kunci dan sertifikat yang ebih kecil. Keuntungan ini sangat penting apabila kita melakukan komputasi pada lingkungan dimana resource komputasi, ruang penyimpanan, bandwidth, atau konsumsi energi yang terbatas.

Elliptic Curve Digital Signature Algorithm (ECDSA) adalah *Elliptic Curve* yang analog dengan *Digital Signature Algorithm* (DSA). ECDSA

pertama kali diajukan pada tahun 1992 oleh Scott Vanstone untuk merespon permintaan NIST (National Institute of Standard and Technology) akan komentar publik terhadap proposal pertama mereka untuk DSS. ECDSA diterima sebagai standard ISO pada tahun 1998, standard ANSI pada 1999, standard IEEE dan FIPS pada tahun 2000.

2. Skema *Digital Signature*

2.1. Latar Belakang

Skema *Digital Signature* dirancang untuk menyediakan persamaan dari tandatangan konvensional (tulisan tangan), bahkan memberikan fungsi lebih. Angka *Digital Signature* bergantung pada angka rahasia yang hanya diketahui oleh pemberi *signature* (kunci privat) dan isi pesan yang diberi tandatangan. *Signature* harus dapat diverifikasi. Apabila terjadi perselisihan, maka tandatangan harus dapat diverifikasi tanpa memerlukan akses ke kunci privat. Perselisihan bisa terjadi apabila orang yang menandatangani pesan berusaha menyangkal bahwa ia telah menandatangani pesan tersebut atau apabila ada orang yang berusaha mengklaim pesan tersebut miliknya.

Pada makalah ini hanya akan dibahas skema *asymmetric Digital Signature* yang berarti bahwa untuk setiap kunci privat, terdapat kunci publik yang bersesuaian. Tiap entitas menjaga kerahasiaan kunci privat miliknya yang digunakan untuk menandatangani pesan, dan mempublikasikan kunci privat yang bersesuaian yang berfungsi untuk melakukan verifikasi tandatangan. Penyandian tidak dilakukan pada pesan, melainkan pada message digest yang dihasilkan dengan menjalankan fungsi hash terhadap pesan.

Idealnya, skema *Digital Signature* tidak dapat dipecahkan dengan serangan *chosen-message*. Ini memastikan bahwa siapapun yang berhasil mendapatkan *signature* suatu entitas A pada sembarang pesan, tidak dapat memecahkan *signature* entitas A pada pesan yang lain.

Skema *Digital Signature* dapat digunakan untuk hal-hal sebagai berikut :

a. Kerahasiaan Pesan
b. Integritas Data (*Data integrity*)
   Memastikan bahwa data tidak diubah oleh orang yang tidak berhak atau dengan cara-cara yang tidak diketahui.
c. Otentikasi
   Memastikan bahwa data bersumber dari pihak yang benar, sesuai dengan yang tertera.
d. Anti Penyangkalan (*Non-Repudiation*)
   Memastikan bahwa suatu entitas tidak dapat menyangkal aksi sebelumnya yang tertulis pada pesan.

Skema *Digital Signature* umumnya digunakan pada protokol kriptografi yang menyediakan layanan lain termasuk otentikasi entitas, otentikasi pertukaran kunci, dan lain sebagainya.

Skema *Digital Signature* yang digunakan saat ini dapat diklasifikasikan berdasarkan skema matematis yang digunakan sebagai basis keamanannya yaitu :

a. Skema *Integer Factorization* (Pemfaktoran bilangan bulat)
   Skema ini mendasarkan aspek keamanan pada sulitnya memfaktorkan bilangan bulat besar menjadi faktor primanya. Contohnya adalah skema RSA dan Rabin.
b. Skema *Discrete Logarithm* (Logaritma Diskrit)

Skema ini mendasarkan aspek keamanannya pada sulitnya memecahkan masalah logaritma diskrit biasa pada sebuah *Finite Field*. Contohnya adalah skema ElGamal, Schnorr, DSA, dan Nyberg-Rueppel.

c. Skema *Elliptic Curve*
Skema ini mendasarkan aspek keamanannya pada sulitnya memecahkan *Elliptic Curve Discrete Logarithm Problem*.

2.2. *Digital Signature Algorithm* (DSA)

DSA diperkenalkan pada Agustus 1991 oleh NIST dan dispesifikasikan dalam FIPS yang disebut *Digital Signature* Standard (DSS). DSA adalah varian dari skema ElGamal. Kekuatan keamanannya terletak pada sulitnya memecahkan masalah logaritma diskrit.

Parameter DSA adalah sebagai berikut :

a. p, adalah bilangan prima dengan panjang L bit, yang dalam hal ini $512 \le L \le 1024$ dan L harus kelipatan 64.
Parameter p bersifat publik dan dapat digunakan bersama-sama oleh orang di dalam kelompok.

b. q, bilangan prima 160 bit, merupakan faktor dari $p - 1$. Dengan kata lain, $(p - 1) \bmod q = 0$. Parameter q berisfat publik.

c. $g = h^{(p-1)/q} \bmod p$, yang dalam hal ini $h < p - 1$ sedemikian sehingga $h^{(p-1)/q} \bmod p > 1$. Parameter g bersifat publik.

d. x, adalah bilangan bulat kurang dari q. Parameter x adalah kunci privat.

e. $y = g^x \bmod p$, adalah kunci publik.

f. m, pesan yang akan diberi tanda-tangan.

Pembangkitan pasangan kunci DSA sebagai berikut :

a. Pilih bilangan prima p dan q, yang dalam hal ini $(p - 1) \bmod q = 0$.

b. Hitung $g = h^{(p-1)/q} \bmod p$, yang dalam hal ini $1 < h < p - 1$ dan $h^{(p-1)/q} \bmod p > 1$.

c. Tentukan kunci privat x, yang dalam hal ini $x < q$.

d. Hitung kunci publik $y = g^x \bmod p$.

Jadi, prosedur di atas menghasilkan:

kunci publik dinyatakan sebagai (p, q, g, y);

kunci privat dinyatakan sebagai (p, q, g, x).

Proses pemberian tandatangan :

a. Ubah pesan m menjadi *message digest* dengan fungsi *hash* SHA, H.

b. Tentukan bilangan acak $k < q$.

c. Tanda-tangan dari pesan m adalah bilangan r dan s. Hitung r dan s sebagai berikut:

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

$$s = (k^{-1} (H(m) + x * r)) \bmod q$$

d. Kirim pesan m beserta tanda-tangan r dan s.

Otentikasi tandatangan :

a. Hitung

$$w = s^{-1} \bmod q$$

$$u_1 = (H(m) * w) \bmod q$$

$$u_2 = (r * w) \bmod q$$

$$v = ((g^{u1} * y^{u2}) \bmod p) \bmod q)$$

b. Jika $v = r$, maka tanda-tangan sah, yang berarti bahwa pesan masih asli dan dikirim oleh pengirim yang benar.

Karena r dan s masing-masing adalah bilangan bulat kurang dari q, *signature* DSA berukuran 320 bit. Keamanan DSA bergantung pada dua *Discrete Logarithm Problem* yang terpisah namun berkaitan. DLP yang pertama memakan waktu komputasi
$O(\exp((c+o(1))(\ln p)^{1/3}(\ln \ln p)^{2/3}))$
dimana $c \approx 1.923$ dan ln n melambangkan fungsi logaritma natural. Jika p adalah bilangan prima berukuran 1024 bit, maka notasi diatas merepresentasikan nilai komputasi yang mustahil dipecahkan. Oleh karena itu, DSA dengan p bilangan prima sepanjang 1024 bit tahan terhadap serangan. DLP yang kedua adalah apabila diketahui p, q, g, dan y, bagaimana menemukan x sedemikian sehingga $y \equiv g^x \pmod p$. Untuk p berukuran besar (misalnya 1024 bit), algoritma terbaik yang pernah diketahui memakan waktu komputasi $\sqrt{\Pi q/2}$. Jika $q \approx 2^{160}$, maka notasi tersebut menghasilkan nilai komputasi yang mustahil dipecahkan. Namun, tingkat keamanan DSA ditentukan oleh ukuran p dan ukuran q. Peningkatan ukuran salah satu parameter tanpa diimbangi dengan peningkatan ukuran parameter lainnya tidak akan menghasilkan peningkatan keamanan yang efektif. Apabila ditemukan algoritma yang lebih baik untuk memecahkan salah satu DLP tersebut, dapat memperlemah keamanan DSA.

Sebagai respon dari draft pertama, FIPS menentukan sebuah metode untuk membangkitkan p dan q agar benar-benar acak. Fitur ini dapat mencegah pembangkitan p dan q yang lemah, yang dapat membuat pemecahan DLP menjadi mudah. FIPS juga membuat metode untuk membangkitkan kunci privat secara acak semu dan membangkitkan kunci sesi k berbasis DES dan SHA-1.

3. *Finite Fields*

Finite Field terdiri atas finite set of elements F bersama denga dua operator biner terhadap F yang disebut *Addition* dan *Multiplication* yang mencakup beberapa properti aritmatika. Orde dari sebuah *Finite Field* adalah jumlah elemen pada *field* tersebut. Sebuah *Finite Field* dengan orde q dinotasikan dengan F_q. Banyak cara dapat digunakan untuk merepresentasikan elemen F_q. Beberapa representasi bisa membuat implementasi dalam *hardware* maupun *software* menjadi lebih efisien.

Jika $q = p^m$ dimana p adalah bilangan prima dan m adalah bilangan bulat positif, maka p disebut karakteristik dari F_q, dan m disebut derajat perpangkatan dari F_q. Kebanyakan standard yang menggunakan teknin *Elliptic Curve Cryptography* membatasi orde *Finite Field* berupa bilangan prima atau perpangkatan dari 2 ($q = 2^m$).

3.1. *Finite Field* F_p

Diketahui p bilangan prima. *Finite Field* F_p disebut prime *field* yang terdiri atas kumpulan bilangan bulat (0, 1, 2, ..., p-1) dengan operasi aritmatika sebagai berikut :

a. *Addition*
   Jika $a, b \in F_p$, maka $a + b = r$, dimana r adalah sisa apabila a + b dibagi dengan p dan $0 \le r \le p-1$. Operasi ini dikenal dengan *Addition* modulo p.
b. *Multiplication*
   Jika $a, b \in F_p$, maka $a * b = s$, dimana s adalah sisa apabila a * b dibagi dengan p dan $0 \le s \le p-1$.

Operasi ini dikenal dengan *Multiplication* modulo p.

c. *Inversion*
Jika elemen tidak kosong dalam F_p, inverse dari a modulo p, dilambangkan dengan a^{-1}, adalah bilangan bulat c ε F_p dimana a*c = 1.

Contoh.
Finite Field F_{23}. Elemennya adalah {0, 1, 2, ... , 22}. Contoh operasi aritmatika : 12 + 20 = 9; 8 * 9 = 3; 8^{-1} = 3.

3.2. *Finite Field* F_{2^m}

Finite Field F_{2^m} disebut characteristic two *Finite Field* atau binary *Finite Field*, dapat dipandang sebagai ruang vektor berdimensi m pada *field* F_2, yang mengandung dua elemen yaitu 0 dan 1. Ada m buah elemen a_0, a_1, ..., a_{m-1} dalam F_{2^m} sedemikian hingga tiap elemen a ε F_{2^m} dapat dituliskan dalam bentuk :
$a = a_0a_0 + a_1a_1 + ... + a_{m-1}a_{m-1}$, dimana a_i ε {0,1}.

Kumpulan $\{a_0, a_1, \ldots, a_{m-1}\}$ disebut basis dari F_{2^m} atas F_2. Jika diberikan basis tersebut, *field* element a dapat direpresentasikan sebagai sebuah bist string $(a_0a_1\ldots a_{m-1})$. Penambahan (*Addition*) dapat dilakukan dengan operasi bitwise XOR. Aturan *Multiplication* bergantung pada basis yang dipilih.

3.2.1. Representasi basis polinomial

Diketahui $f(x) = x^m + f_{m-1}x^{m-1} + ... + f_2x^2 + f_1x + f_0$ (dimana f_i ε {0, 1} untuk i = 0, 1, ..., m-1) adalah fungsi polinomial yang tidak dapat direduksi derajat m pada F_2. f(x) tidak dapat difaktorkan menjadi dua buah fungsi polinomial dengan derajat kurang dari m pada F_2. f(x) disebut *reduction polynomial.*

$F_{2^m} = \{a_{m-1}x^{m-1} + ... + a_1x + a_0 : a_i \varepsilon \{0,1\}\}$
Finite element $a_{m-1}x^{m-1} + ... + a_1x + a_0$ dinotasikan dalam bit string $(a_{m-1}\ldots a_1a_0)$ sepanjang m sehingga :
$F_{2^m} = \{(a_{m-1}\ldots a_1a_0) : a_i \varepsilon \{0,1\}\}$
Elemen identitas perkalian (1) dinotasikan dengan bit string (00…01) sedangkan elemen identitas penjumlahan dinotasikan dengan (00...00).

Apabila menggunakan representasi basis polinomial, operasi aritmatika yang berlaku sebagai berikut :

a. *Addition*
Jika $a = (a_{m-1}\ldots a_1a_0)$ dan $b = (b_{m-1}\ldots b_1b_0)$ adalah elemen dari F_{2^m} maka $a + b = c = (c_{m-1}\ldots c_1c_0)$ dimana $c_i = (a_i + b_i) \bmod 2$. *Addition* dilakukan dalam operasi bitwise.

b. *Multiplication*
Jika $a = (a_{m-1}\ldots a_1a_0)$ dan $b = (b_{m-1}\ldots b_1b_0)$ adalah elemen dari F_{2^m} maka $a * b = r = (r_{m-1}\ldots r_1r_0)$, dimana polinomial $r_{m-1}x^{m-1} + ... + r_1x + r_0$ adalah sisa apabila polinomial $(a_{m-1}x^{m-1} + ... + a_1x + a_0) * (b_{m-1}x^{m-1} + ... + b_1x + b_0)$ dibagi dengan f(x) pada F_2.

c. *Inversion*
Jika a adalah elemen bukan 0 dari F_{2^m}, inverse dari a, dinotasikan dengan a^{-1}, adalah elemen unik c ε F_{2^m} dimana a*c = 1.

Contoh :
Representasi basis polinomial *Finite Field* F_{2^4}. Diketahui $f(x) = x^4 + x + 1$ adalah *reduction polynomial*. Maka 16 elemen dari F_{2^4} adalah :

0(0000)	x^3(1000)
1(0001)	$x^3 + 1$(1001)

x(0010)	x^3 + x(1010)
x + 1 (0011)	x^3 + x + 1(1011)
x^2(0100)	x^3 + x^2(1100)
x^2 + 1(0101)	x^3 + x^2 + 1(1101)
x^2 + x(0110)	x^3 + x^2 + x(1110)
x^2 + x + 1(0111)	x^3 + x^2 + x + 1(1111)

Contoh operasi aritmatika pada F_{2^4} :

- (1101) + (1001) = (0100)
- (1101) * (1001) = (1111)
- $(1101)^{-1}$ = (0100)

Sebuah trinomial pada F_2 adalah polinomial dengan bentuk $x^m + x^k + 1$, dimana $1 \leq k \leq m-1$. Sebuah pentanomal pada F_2 adalah polinomial dengan bentuk $x^m + x^{k3} + x^{k2} + x^{k1} + 1$, dimana $1 \leq k1 < k2 < k3 \leq m-1$. ANSI membuat beberapa peraturan untuk memilih reduction polynomial untuk merepresentasikan elemen dari F_{2^m} :

a. Jika ada trinomial yang tidak dapat direduksi (*irreductible trinomial*) berderajat m pada F_2, maka reduction polynomial f(x) pasti adalah sebuah irreductible trinomial berderajat m pada F_2.
b. Jika tidak ada *irreducible trinomial* berderajat m pada F_2, maka *reduction polynomi*al f(x) pasti sebuah irreducible pentanomial berderajat m pada F_2.

3.2.2. Representasi basis normal

Basis normal dari F_{2^m} pada F_2 adalah basis dengan bentuk $\{\beta, \beta^2, \beta^{2^2}, ..., \beta^{2^{m-1}}\}$, dimana $\beta \varepsilon F_{2^m}$. Basis yang demikian selalu ada. Elemen apapun dimana $a \varepsilon F_{2^m}$ dapat dituliskan sebagai $a = \sum_{i=0}^{m-1} a_i \beta^{2^i}$, dimana $a_i \varepsilon \{0,1\}$. Representasi basis normal memiliki keunggulan komputasi yaitu operasi perpangkatan dapat dilakukan secara efisien, namun operasi perkalian menjadi kurang efisien. Karena alasan inilah, ANSI menspesifikasikan penggunaan *Gaussian normal bases* (GNB) yang dengannya operasi perkalian menjadi lebih sederhana dan efisien.

Tipe dari sebuah GNB adalah sebuah bilangan bulat positif yang menunjukkan kompleksitas operasi perkalian relatif terhadapa basis. Semakin kecil tipe GNB, semakin efisien perkalian yang dilakukan.

Sebuah GNB ada ketika m tidak dapat dibagi 8. Jika diketahui m sebuah bilangan bulat positif yang tidak dapat dibagi 8, dan T adalah bilangan bulat positif. Maka sebuah GNB tipe T untuk F_{2^m} ada jika dan hanya jika p = Tm + 1 adalah bilangan prima dan pbb(Tm/k,m) = 1, dimana k adalah orde perkalian dari 2 modulo p.

Jika $\{\beta, \beta^2, \beta^{2^2}, ..., \beta^{2^{m-1}}\}$ adalah basis normal dari F_{2^m} terhadap F_2, maka elemen $a = \sum_{i=0}^{m-1} a_i \beta^{2^i}$ direpresentasikan dengan string biner $(a_0a_1...a_{m-1})$ dengan panjang m sehingga $F_{2^m} = \{(a_0a_1...a_{m-1}) : a_i \varepsilon \{0,1\}\}$.

Elemen identitas perkalian (1) direpresentasikan dengan bitstring semua bernilai 1, dan elemen identitas penjumlahan (0) direpresentasikan dengan bitstring semua bernilai 0. Operasi aritmatika yang berlaku untuk GNB tipe T adalah :

a. *Addition*
   Jika $a = (a_0a_1...a_{m-1})$ dan $b = (b_0b_1...b_{m-1})$ adalah elemen dari F_{2^m}, maka $a + b = c = (c_0c_1...c_{m-1})$ dimana $c_i = (a_i + b_i) \bmod 2$. Penjumlahan dilakukan secara bitwise.
b. *Squaring*

Jika a = $(a_0a_1...a_{m-1})$ adalah elemen dari F_{2^m}, maka $a^2 = \left(\sum_{i=0}^{m-1} a_i \beta^{2^i}\right)^2 = \sum_{i=0}^{m-1} a_i \beta^{2^{i+1}} = \sum_{i=0}^{m-1} a_{i-1} \beta^{2^i} = (a_{m-1}a_0a_1 \ldots a_{m-2})$

c. *Multiplication*
Jika p =Tm + 1, dan u ε F_p adalah elemen orde T. Definisikan barisan F(1), F(2), ... , F(p-1) dengan cara $F(2^i u^j \bmod p) = i$ untuk $0 \le i \le m-1$, $0 \le j \le T-1$. Jika a = $(a_0a_1...a_{m-1})$ dan b = $(b_0b_1...b_{m-1})$ adalah elemen dari F_{2^m}, maka a * b = c = $(c_0c_1...c_{m-1})$ dimana

$$c_l = \begin{cases} \sum_{k=1}^{p-2} a_{F(k+1)+l} b_{F(p-k)+l} \\ \sum_{k=1}^{m/2} \left(a_{k+l-1} b_{m/2+k+l-1} + a_{m/2+k+l-1} b_{k+l-1}\right) \\ \quad + \sum_{k=1}^{p-2} a_{F(k+1)+l} b_{F(p-k)+l} \end{cases}$$

$0 \le l \le m-1$.

d. *Inversion*
Jika a adalah elemen tidak nol dari F_{2^m}, inverse a dalam F_{2^m} dinotasikan a^{-1} adalah elemn unik c ε F_{2^m} dimana a*c = 1.

ANSI menspesifikasikan aturan untuk memilih sebuah GNB sebagai berikut :

a. Jika ada GNB tipe 2 pada F_{2^m} maka basis ini harus digunakan.
b. Jika tidak ada GNB basis 2 pada F_{2^m} namun ada GNB basis 1, maka GNB basis 1 harus digunakan.
c. Jika tidak ada keduanya, maka GNB dengan basis terkecil yang harus digunakan.

4. *Elliptic Curve* pada *Finite Fields*

4.1. *Elliptic Curve* pada F_p

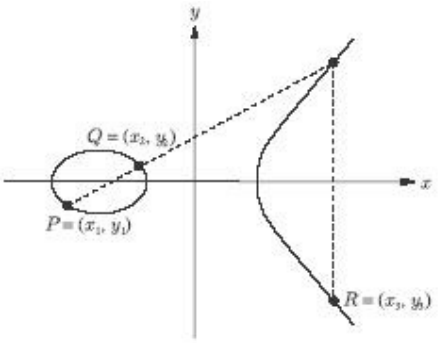
Jika p>3 adalah bilangan prima. Sebuah *Elliptic Curve* E pada F_p didefinisikan sebagai persamaan :
$y^2 = x^3 + ax + b$
dimana a,b elemen F_p dan $4a^3 + 27b^2 \not\equiv 0 \pmod p$. Set $E(F_p)$ mengandung semua titik (x,y), x elemen F_p, y elemen F_p yang memenuhi persamaan elips diatas bersama titik O yang disebut point of infinity.

Contoh :
Elliptic Curve pada F_{23}. Diketahui p = 23 dan *Elliptic Curve* E : $y^2 = x^3 + x + 4$. Didapatkan bahwa $4a^3 + 27b^2 = 4 + 432 = 436 \equiv 22 \pmod{23}$, sehingga E adalah *Elliptic Curve*. Titik-titik pada $E(F_{23})$ adalah O dan :

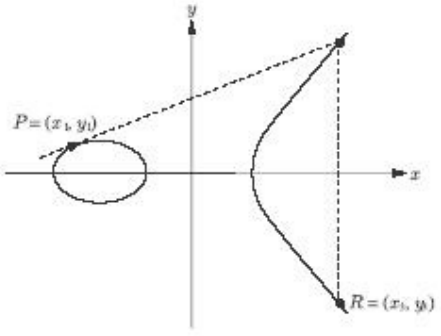
(0,2)	(0,21)	(1,11)	(1,12)	(4,7)
(4,16)	(7,3)	(7,20)	(8,8)	(8,15)
(9,11)	(9,12)	(10,5)	(10,18)	(11,9)
(11,14)	(13,1)	(13,22)	(14,5)	(14,18)
(15,6)	(15,17)	(17,9)	(17,14)	(18,9)
(18,7)	(22,1)	(22,19)		

Terdapat aturan yang disebut chord-and-tangent rule, untuk menjulahkan dua titik pada *Elliptic Curve* $E(F_p)$ untuk mendapatkan titik ketiga pada *Elliptic Curve*. Bersama dengan operasi penjumlahan ini, terdapat kumpulan titik pada $E(F_p)$ yang membentuk kelompok dengan O sebagai identitasnya. Kelompok inilah yang digunakan dalam membangun *Elliptic Curve cryptosystem*.

Aturan penjumlahan dijelaskan melalui gambar berikut :



Diketahui P = (x_1, y_1) dan Q = (x_2, y_2) adalah dua titik pada *Elliptic Curve* E. Maka penjumlahan P dan Q dinotasikan sebagai R = (x_3, y_3), didefinisikan sebaai berikut. Pertama, gambar sebuah garis melalui P dan Q yang memotong *Elliptic Curve* pada titik ketiga. R adalah pencerminan dari titik ketiga tersebut terhadap sumbu x.



Jika P = (x_1, y_1), maka dua kali P, dilambangkan sebagai R = (x_3, y_3) didefinisikan sebagai berikut. Pertama gambar sebuah garis tangent dari titik P yang memotong *Elliptic Curve* pada titik kedua. R adalah pencerminan dari titik kedua tersebut terhadap sumbu x.

Dari deskripsi diatas dapat dituliskan formula aljabar sebagai berikut :

a. P + O = O + P = P untuk semua P elemen dari $E(F_p)$.
b. Jika P = (x,y) elemen dari $E(F_p)$, maka (x,y) + (x,-y) = O. (Titik (x, -y) dinotasikan sebagai –P, dan disebut negatif dari P).
c. Jika P = (x_1, y_1) elemen dari $E(F_p)$ dan Q = (x_2, y_2) elemen dari $E(F_p)$, dimana P ≠ ±Q, maka P + Q = (x_3, y_3) dimana

$$x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2 \text{ dan}$$

$$y_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3) - y_1$$

d. Jika P = (x_1, y_1) elemen dari $E(F_p)$ dimana P ≠ -P maka 2P = (x_3, y_3) dimana

$$x_3 = \left(\frac{3x_1^2 + a}{2y_1}\right)^2 - 2x_1 \text{ dan}$$

$$y_3 = \left(\frac{3x_1^2 + a}{2y_1}\right)(x_1 - x_3) - y_1$$

4.2. *Elliptic Curve* pada F_{2^m}

Sebuah *Elliptic Curve* E pada F_{2^m} didefinisikan dalam persamaan
$y^2 + xy = x^3 + ax^2 + b$
dimana a,b elemen F_{2^m} dan b ≠ 0. Set E(F_{2^m}) terdiri atas semua titik (x,y), x elemen F_{2^m}, y elemen F_{2^m} yang memenuhi persamaan elips diatas, bersama dengan titik khusus O yang disebut point of infinity.

Contoh:
Elliptic Curve pada F_{2^4}. Diketahui F_{2^4} direpresentasikan sebagai irreducible trinomial $f(x) = x^4+x+1$. Diketahui *Elliptic Curve* E : $y^2 + xy = x^3 + \alpha^4x^2 + 1$. b ≠ 0, sehingga E adalah sebuah *Elliptic Curve*.Titik-titik pada E(F_{2^4}) adalah O dan titik-titik sebagai berikut :

(0,1)	(1, α^5)	(1, α^6)	(α^3, α^{13})	(α^3, α^8)
(α^5, α^3)	(α^5, α^9)	(α^6, α^{11})	(α^6, α^8)	(α^9, α^{14})
(α^9, α^{10})				

α^{13}) (α^{10}, α) (α^{10}, α^8) (α^{12}, 0) (α^{12}, α^{12})

Sebagaimana pada F_p, pada F_{2^m} juga terdapat chord-and-tangent rule. Formula aljabar untuk F_{2^m} adalah sebagai berikut :

a. P + O = O + P = P untuk semua P elemen E(F_{2^m}).
b. Jika P = (x, y) elemen E(F_{2^m}), maka (x, y) + (x, x + y) = O.(Titik (x, x + y) dinotasikan sebagai –P, dan disebut negatif dari P).
c. Jika P = (x_1, y_1) elemen dari E(F_{2^m}) dan Q = (x_2, y_2) elemen dari E(F_{2^m}), dimana P ≠ ±Q, maka P + Q = (x_3, y_3) dimana

$$x_3 = \left(\frac{y_1 + y_2}{x_1 + x_2}\right)^2 + \frac{y_1 + y_2}{x_1 + x_2} + x_1 + x_2 + a$$

dan

$$y_3 = \left(\frac{y_1 + y_2}{x_1 + x_2}\right)(x_1 + x_3) + x_3 + y_1$$

d. Jika P = (x_1, y_1) elemen dari E(F_{2^m}) dimana P ≠ -P maka 2P = (x_3, y_3) dimana

$$x_3 = x_1^2 + \frac{b}{x_1^2} \text{ dan}$$

$$y_3 = x_1^2 + \left(x_1 + \frac{y_1}{x_1}\right)x_3 + x_3$$

Contoh :
Menggunakan data sebelumnya.

a. P = (α^6, α^8) dan Q = (α^3, α^{13}). Maka P + Q = (x_3, y_3) dihitung sebagai berikut :

$$x_3 = \left(\frac{\alpha^8 + \alpha^{13}}{\alpha^6 + \alpha^3}\right)^2 + \frac{\alpha^8 + \alpha^{13}}{\alpha^6 + \alpha^3} + \alpha^6 + \alpha^3 + \alpha^4$$

$$= \left(\frac{\alpha^3}{\alpha^2}\right)^2 + \frac{\alpha^3}{\alpha^2} + \alpha^6 + \alpha^3 + \alpha^4 = 1$$

dan

$$y_3 = \left(\frac{\alpha^8 + \alpha^{13}}{\alpha^6 + \alpha^3}\right)^2 (\alpha^6 + 1) + 1 + \alpha^8$$

$$= \left(\frac{\alpha^3}{\alpha^2}\right)(\alpha^{13}) + \alpha^2 = \alpha^{13}$$

Sehingga P + Q = (1, α^{13})

b. P = (α^6, α^8). Maka 2P = P + P = (x_3, y_3) dihitung sebagai berikut :

$$x_3 = (\alpha^6)^2 + \frac{1}{(\alpha^6)^2}$$

$$= \alpha^{12} + \alpha^3 = \alpha^{10}$$

dan

$$y_3 = (\alpha^6)^2 + \left(\alpha^6 + \frac{\alpha^8}{\alpha^6}\right)\alpha^{10} + \alpha^{10}$$

$$= \alpha^{12} + \alpha^{13} + \alpha^{10} = \alpha^8$$

Sehingga 2P = (α^{10}, α^8)

5. Domain parameter ECDSA

Domain parameter pada ECDSA terdiri atas :

a. Ukuran *field* q, dimana q = p (bilangan prima), atau $q = 2^m$.
b. Indikator FR (*Field Representation*) adalah representasi yang digunakan untuk elemen dari F_q.
c. (optional) Sebuah bit string seedE dengan panjang minimal 160 bit.
d. Dua elemen *field* a dan b pada F_q yang mendefinisikan persamaan *Elliptic Curve* E pada F_q.
e. Dua elemen *field* x_G dan y_G pada F_q yang mendefinisikan finite point G = (x_G, y_G) pada E(F_q).
f. Orde n dari titik G, dengan $n > 2^{160}$ dan $n > 4\sqrt{q}$.
g. Kofaktor h = #E(F_q)/n

Berikut adalah metode untuk membangkitkan domain parameter yang aman secara kriptografi :

a. Pilih koefisien a dan b dari F_q secara acak. E adalah kurva $y^2 = x^3 + ax + b$ untuk kasus q = p, dan y^2

$+ xy = x^3 + ax^2 + b$ pada kasus $q = 2^m$.

b. Hitung $N = \#E(F_q)$
c. Verifikasi bahwa N dapat dibagi dengan bilangan prima besar n ($n > 2^{160}$ dan $n > 4\sqrt{q}$). Jika tidak kembali ke langkah a.
d. Verifikasi bahwa n tidak habis membagi $q^k - 1$ untuk tiap k, dimana $1 \leq k \leq 20$. Jika tidak kembali ke langkah a.
e. Verifikasi bahwa $n \neq q$. Jika tidak kembali ke langkah a.
f. Pilih sembarang titik G' elemen $E(F_q)$ dan set $G = (N/n)G'$. Ulangi hingga $G \neq O$.

Algoritma berikut dapat digunakan untuk menguji validitas domain parameter D.

Input : Domain parameter D = (q, FR, a, b, G, n, h).

Output : Penerimaan atau penolakan validitas D.

a. Verifikasi bahwa q adalah bilangan prima (q = p) atau perpangkatan dari 2 ($q = 2^m$).
b. Verifikasi bahwa FR adalah representasi yang valid untuk F_q.
c. Verifikasi bahwa $G \neq O$.
d. Verifikasi bahwa a, b, x_G, dan y_G adalah representasi yang benar dari elemen F_q (sebagai contoh bilangan bulat pada interval [0, p-1] pada kasus q = p, dan bit string dengan panjang m bit pada kasus $q = 2^m$).
e. Verifikasi bahwa a dan b mendefinisikan *elliptic curve* pada F_q (misalnya $4a^3 + 27b^2 \not\equiv 0 \pmod p$ jika q = p; $b \neq 0$ jika $q = 2^m$).
f. Verifikasi bahwa G terletak pada *elliptic curve* yang didefinisikan oleh a dan b (misalnya $y_G^2 = x_G^3 + ax_G + b$ pada kasus q = p dan $y_G^2 + x_G y_G = x_G^3 + ax_G^2 + b$ pada kasus $q = 2^m$).
g. Verifikasi bahwa n adalah bilangan prima.
h. Verifikasi bahwa nG = O.
i. Hitung $h' = \left\lfloor \left(\sqrt{q}+1\right)^2 / n \right\rfloor$ dan verifikasi bahwa h = h'.
j. Verifikasi bahwa n tidak habis membagi q^k-1 untuk tiap k dimana $1 \leq k \leq 20$.
k. Verifikasi bahwa $n \neq q$.
l. Jika ada diantara verifikasi diatas yang gagal, maka D tidak valid. Jika sebaliknya, maka D valid.

Jaminan bahwa D = (q, FR, a, b, G, n, h) adalah domain *elliptic curve* yang valid dapat diperoleh oleh suatu entitas dengan menerapakan salah satu dari metode berikut :

a. A melakukan validasi domain parameter menggunakan algoritma diatas.
b. A membangkitkan D sendiri menggunakan sistem yang terpercaya.
c. A menerima jaminan dari pihak terpercaya T (misalnya Certification Authority) bahwa T telah melakukan validasi D menggunakan algortima diatas.
d. A menerima jaminan dari pihak terpercaya T bahwa D dibangkitkan menggunakan sistem yang terpercaya.

6. Pasangan Kunci ECDSA

6.1. Pembangkitan Pasangan Kunci

Pasangan kunci berasosiasi dengan *Elliptic Curve* domain parameter D = (q, FR, a, b, G, n, h). Setiap entitas harus memastikan bahwa domain parameter yang digunakan valid sebelum pembangkitan kunci.

Pembangkitan kunci dilakukan sebagai berikut :

a. Pilih bilangan bulat d secara acak atau acak semu pada interval [1, n-1].
b. Hitung Q = dG.
c. Kunci publik adalah Q, dan kunci privat adalah d.

6.2. Validasi Kunci Publik

Validasi kunci publik dilakukan untuk memastikan bahwa kunci publik memiliki properti aritmatika yang diperlukan. Alasan melakukan validasi kunci publik adalah mencegah adanya kunci publik yang tidak valid yang memungkinkan terjadinya serangan serta mendeteksi kesalahan pada saat transmisi kunci publik.

Algoritma yang dapat digunakan untuk mengecek validitas kunci publik adalah sebagai berikut :
Input : kunci publik Q = (x_Q, y_Q) yang berasosiasi dengan domain parameter valid (q, FR, a, b, G, n, h).
Output : Penerimaan atau penolakan atas validitas Q.

a. Cek bahwa $Q \neq O$.
b. Cek bahwa $x_Q$dan y_Q adalah representasi yang benar dari elemen F_q (sebagai contoh bilangan bulat pada interval [0, p-1] pada kasus q = p, dan bit string dengan panjang m bit pada kasus $q = 2^m$)
c. Cek bahwa Q berada pada *Elliptic Curve* yang didefinisikan oleh a dan b.
d. Cek bahwa nQ = O.
e. Jika ada pengecekan yang gagal, maka Q tidak valid. Jika sebaliknya, maka Q valid.

Suatu entitas A dapat memastikan bahwa kunci publik Q valid dengan metode :

a. A melakukan validasi kunci publik menggunakan algoritma diatas.
b. A membangkitkan Q sendiri menggunakan sebuah sistem terpercaya.
c. A menerima jaminan dari pihak terpercaya T (misalnya Certification Authority) bahwa T telah melakukan validasi kunci publik menggunakan algortima diatas.
d. A menerima jaminan dari pihak terpercaya T bahwa Q dibangkitkan menggunakan sistem yang terpercaya.

6.3. Pembuktian Kepemilikan Kunci Privat

Jika suatu entitas C dapat mensertifikasi kunci publik A seperti kunci publiknya sendiri, maka C dapat mengklaim bahwa pesan yang ditandatangani A aslinya dari C. Untuk menghindari hal ini, CA harus mewajibkan semua entitas A untuk membuktikan kepemilikan kunci privat yang berkorespondensi dengan kunci publik sebelum CA mensertifikasi kunci publik milik A. Bukti kepemilikan ini dapat dipenuhi dengan beberapa cara, misalnya dengan mengharuskan A menandatangani pesan sesuai pilihan CA.

7. Pembangkitan dan Verifikasi Tandatangan ECDSA

Untuk menandatangani pesan m, suatu entitas A dengan domain parameter D = (q, FR, a, b, G, n, h) dan pasangan kunci (d, Q) melakukan hal-hal sebagai berikut :

a. Pilih suatu bilangan bulat k secara acak atau acak semu dimana $1 \leq k \leq n-1$.
b. Hitung kG = (x_1, y_1) dan ubah x_1 menjadi bilangan bulat.
c. Hitung $r = x_1 \bmod n$. Jika r = 0 kembali ke langkah a.
d. Hitung $k^{-1} \bmod n$.
e. Hitung SHA-1(m) dan ubah bit string hasilnya menjadi bilangan bulat e.
f. Hitung $s = k^{-1}(e + dr) \bmod n$. Jika s = 0 kembali ke langkah a.
g. Tandatangan untuk pesan m adalah (r, s).

Untuk melakukan verifikasi tandatangan A (r, s) pada pesan m, B

mengambil copy otentik dari domain parameter A D = (q, FR, a, b, G, n, h) dan kunci publik Q yang berasosiasi. B melakukan hal-hal sebagai berikut :

a. Pastikan bahwa r dan s adalah bilangan bulat pada interval [1, n-1].
b. Hitung SHA-1(m) dan ubah bit string hasilnya menjadi bilangan bulat e.
c. Hitung $w = s^{-1} \bmod n$.
d. Hitung $u_1 = ew \bmod n$ dan $u_2 = rw \bmod n$.
e. Hitung $X = u_1G + u_2Q$.
f. Jika X = O, tolak tandatangan. Sebaliknya, ubah koordinat x_1 dari X menjadi bilangan bulat χ. Lalu hitung $v = \chi \bmod n$.
g. Terima tandatangan jika dan hanya jika v = r.

Jika *signature* (r, s) pada pesan m benar-benar dibangkitkan oleh A, maka $s = k^{-1}(e + dr) \bmod n$. Jika diturunkan menghasilkan :
$K \equiv s^{-1}(e + dr) \equiv s^{-1}e + s^{-1}rd \equiv we + wrd \equiv u_1 + u_2d \pmod n$

Sebelum melakukan verifikasi tandatangan A pada pesan, B perlu memiliki copy otentik dari domain parameter milik A yaitu D dan kunci publik Q yang bersesuaian. Kunci publik otentik umumnya didistribusikan melalui sertifikat digital. Sertifikat digital pada kunci publik A harus mengandung informasi yang dapat mengidentifikasi A (nama dan alamat), domain parameter D milik A, kunci publik Q, dan tandatangan CA pada informasi ini.

8. Pertimbangan Keamanan

Tujuan dari ECDSA adalah agar *Digital Signature* yang dibangkitkan tahan terhadap serangan *chosen-message*. *Chosen-message* attack dilakukan dengan mendapatkan dan mempelajari *signature* suatu entitas pada kumpulan pesan hingga dapat memecahkan *signature* entitas yang sama pada pesan yang lain.

Serangan yang mungkin dialami oleh ECDSA apat diklasifikasikan sebagai berikut :

a. Serangan pada *Elliptic Curve Discrete Logarithm Problem*.
b. Serangan pada fungsi hash yang digunakan.
c. Serangan yang lain.

8.1. *Elliptic Curve Discrete Logarithm Problem* (ECDLP)

Salah satu cara bagi penyerang yang dapat sukses adalah menghitung kunci privat dari domain parameter (q, FR, a, b, G, n, h) dan kunci publik Q. Permasalahan ECDLP adalah jika diberikan sebuah *Elliptic Curve* E yang terdefinisi pada sebuah *Finite Field* F_q, sebuah titik P elemen $E(F_q)$ pada orde n, dan sebuah titik Q = lP dimana $0 \leq l \leq n-1$, tentukan l.

8.1.1. Jenis-jenis serangan yang diketahui

Algoritma yang telah diketahui untuk memecahkan ECDLP adalah sebagai berikut :

a. *Naïve exhaustive search*
   Pada metode ini, seseorang melakukan komputasi P (P : P, 2P, 3P, 3P, …) secara exhaustive search (brute force), sampai Q didapatkan. Metode ini dapat memakan waktu n langkah pada kasus terburuk.
b. Algoritma Pohlig-Hellman
   Algoritma ini mengeksploitasi faktorisasi n, dimana n adalah orde dari titik P. Akibat dari ditemukannya algoritma ini, apabla seseorang ingin mendapatkan ECDLP yang sulit dipecahkan, harus digunakan *Elliptic Curve* dengan orde yang

dapat dibagi dengan bilangan prima besar n.

c. Algoritma *baby-step giant-step*
   Algoritma ini memiliki trade-off antara waktu dan memori dibandingkan dengan exhaustive search. Algoritma ini memerlukan storage sebanyak $\sqrt{n}$ titik, dan memerlukan waktu $\sqrt{n}$ langkah pada kasus terburuk.
d. Algoritma Pollard's rho
   Algoritma ini adalah versi acak dari algoritma baby-step giant-step. Algoritma ini membutuhkan waktu hampir sama dengan baby-step giant-step ($\sqrt{\pi m/2}$ langkah) namun membutuhkan storage yang jauh lebih kecil.
e. Algoritma Parallelized Pollard's rho
   Ini adalah penerapan algoritma Pollard's rho pada komputer paralel sebanyak r processor, sehingga running time menjadi $\sqrt{\pi m}/2r$ langkah.
f. Metode Pollard's lambda
   Metode lain yang dikembangkan oleh Pollard. Metode ini juga dapat diparalelkan. Metode ini lebih cepat pada kondisi logaritma bernilai antar interval [0,b] dan [0,n-1], dimana b<0.39n.
g. *Multiple Logarithm*
h. *Supersingular Elliptic Curve*
i. *Prime-field anomalous curves*
j. *Curves defined over a small field*
k. *Curves defined over* F_{2^m}, m bilangan komposit.
l. *Non-applicability of index-calculus methods*
m. *Xedni-calculus attacks*
n. *HyperElliptic Curve*s
o. *Equivalence to other Discrete Logarithm Problem*s

8.2. Serangan pada fungsi hash

Fungsi hash H adalah fungsi yang memetakan bit string dengan panjang tidak tentu ke bit sting dengan panjang tetap t sedemikian sehingga :

a. H dapat dihitung dengan efisien.
b. Untuk semua y elemen $\{0,1\}^t$ tidak mungkin (sangat sulit secara komputasi) untuk menemukan sebuah bit string x sedemikian sehingga H(x) = y. Sifat ini disebut Preimage Resistance. Sifat ini disebut juga fungsi hash satu arah.
c. Tidak mungkin (sangat sulit secara komputasi) untuk menemukan bit string x_1 dan x_2 sedemikian sehingga $H(x_1) = H(x_2)$. Sifat ini disebut Collision Resistance.

Apabila fungsi hash tidak memenuhi sifat preimage resistance dan collision resistance, serangan berikut ini dapat memecahkan ECDSA :

a. Jika SHA-1 tidak memenuhi sifat preimage resistant, E dapat memecahkan *signature* A dengan cara sebagai berikut. E memilih bilangan bulat sembarang l dan menghitung r, dimana x koordinat dari Q + lG dalam modulo n. E mengeset s = r dan menghitung e = rl mod n. Jika E dapat menemukan pesan m sedemikian sehingga e = SHA-1(m), maka (r,s) adalah *signature* yang valid untuk m.
b. Jika SHA-1 tidak memenuhi sifat collision resistant, maka entitas A dapat melakukan penyangkalan *signature* dengan cara sebagai berikut. Pertama A membangkitkan dua pesan m dan m' sedemikian sehingga SHA-1(m) = SHA-1(m'), hal tersebut menunjukkan adanya collision. A lantas menandatangani m namun kemudian menyangkalnya, dan mengaku bahwa ia menandatangani m' (setiap

signature untuk m adalah juga *signature* untuk m').

9. Pertimbangan Implementasi

Sebelum mengimplementasikan ECDSA, beberapa pilihan utama harus dipertimbangkan yaitu :

a. Tipe *Finite Field* yang digunakan (F_p atau F_{2^m}).
b. Representasi *field* (basis polinomial atau normal).
c. Tipe *Elliptic Curve* E pada F_q (random curve atau Koblitz curve).
d. Representasi titik *Elliptic Curve* (affine atau projective).

Banyak faktor yang dapat mempengaruhi pilihan-pilihan diatas. Semuanya harus dipertimbangkan untuk mendapatkan solusi terbaik untuk aplikasi tertentu. Faktor-faktor tersebut antara lain :

a. Pertimbangan keamanan.
b. Kecocokan dengan metode yang ada untuk mengoptimalkan operasi aritmatika pada *Finite Field* (*Addition*, *Multiplication*, *Squaring*, *Inversion*).
c. Kecocokan dengan metode yang ada untuk mengoptimalkan operasi aritmatika pada *Elliptic Curve* (*point Addition*, *point doubling*, dan *scalar Multiplication*).
d. Platform aplikasi (*software*, *hardware*, *firmware*).
e. Batasan lingkungan komputasi (kecepatan processor, storage, ukuran kode, konsumsi energi, memori).
f. Batasan lingkungan komunikasi (bandwidth, response time).

10. Pertimbangan Interoperabilitas

Tujuan standard kriptografi adalah :

a. Untuk memfasilitasi penggunaan teknik kriptografi yang baik dan terspesifikasi dengan baik.
b. Meningkatkan interoperabilitas antar implementasi yang berbeda.

Interoperabilitas dapat ditingkatkan dengan menspesifikasikan secara lengkap langkah-langkah skema kriptografi dan format pertukaran data seperti domain parameter, kunci, dan pertukaran pesan, dan dengan membatasi pilihan yang ada pada saat implementasi. Faktor-faktor yang dapat mempengaruhi interoperabilitas pada ECDSA antara lain :

a. Jumlah dan tipe *Finite Field* yang diizinkan.
b. Jumlah representasi elemen yang diperbolehkan untuk tiap *Finite Field* yang diizinkan.
c. Jumlah *Elliptic Curve* yang diperbolehkan untuk tiap *Finite Field* yang diizinkan.
d. Format elemen *field*, titik pada *Elliptic Curve*, domain parameter, kunci publik, dan *signature*.

11. Kesimpulan

a. ECDSA adalah penerapan lain dari DSA. Kedua algoritma ini sama-sama menggunakan SHA-1 untuk menghasilkan message digest. Bedanya adalah pada enkripsi message digest, ECDSA menggunakan *Elliptic Curve Cryptosystem*, sedangkan DSA menggunakan RSA.
b. ECDSA memiliki tingkat keamanan yang lebih tinggi dibanding DSA dengan panjang kunci yang sama.
c. ECDSA cocok diimplementasikan pada lingkungan yang memiliki keterbatasan resource komputasi, energi, maupun komunikasi seperti pada mobile device dan smart card.

d. Untuk mendapatkan hasil yang efisien, sebelum implementasi, perlu dipikirkan mengenai parameter-parameter yang akan digunakan pada penerapan ECDSA.
e. Perlu adanya standard khusus mengenai penerapan ECDSA untuk meningkatkan interoperabilitas.

12. Referensi

[1] Bassham III, Lawrence E. (2004). *The Elliptic Curve Digital Signature Validation System (ECDSAVS)*. National Institute of Standard and Technology.

[2] Certicom Research. (2000). *Standard for Efficient Cryptography : Elliptic Curve Cryptography.*

[3] Johnson, Don. Menezes, Alfred. Vanstone, Scott. (2001) . *The Elliptic Curve Digital Signature Algorithm*. Certicom Research and Department of Combinatorics and Optimization, University of Waterloo, Canada.

[4] Lopez, Julio. Dahab, Ricardo. (2000). *An Overview of Elliptic Curve Cryptography*. Institute of Computing, State University of Campinas, Brazil.

[5] Munir, Rinaldi. (2006). *Tandatangan Digital – Bahan Kuliah Kriptografi*. Materi Kuliah Kriptografi, Teknik Informatika ITB.