ISSN (e): 2250 – 3005 || Vol, 04 || Issue, 12 || December – 2014 ||
International Journal of Computational Engineering Research (IJCER)

# Implementation of Elliptic Curve Digital Signature Algorithm Using Variable Text Based Message Encryption with Message Digest

**Rajasekhar Bandapalle Mulinti' Dr.G.A.Ramachandra**
[1,2] Research Scholar, Department of Computer Science & Technology, [1,2] ,Associate Professor, Sri Krishnadevaraya University, INDIA

**ABSTRACT:**
*Digital Signatures are considered as digital counterparts to handwritten signatures, and they are the basis for validating the authenticity of a connection. It is well known that with the help of digital signature, forgery of digital information can be identified and it is widely used in e-commerce and banking applications. Elliptic curve digital signatures (ECDSA) are stronger and ideal for constrained environments like smart cards due to smaller bit size, thereby reducing processing overhead. We have implemented ECDSA over Elliptic Curve (EC) P-192 and P-256 using various Text Message encryptions which are Variable Size Text Message(VTM), Fixed Size Text Message(FTM) and Text Based Message(TBM) encryption methods and compared their performance. In the existing Variable Text Based Message used the plain message for generating digital signature but in the new approach, we have converted plain message to digested message using SHA algorithm and then created digital signature which is more faster than existing approach.*

***Keywords:*** *Digital Signature, Elliptic Curve Digital Signature Algorithm, Elliptic Curve Cryptography, ECDLP.*

## I. Introduction

Cryptography is the branch of cryptology dealing with the design of algorithms for encryption and decryption, intended to ensure the secrecy and/or authenticity of message. The Digital Signature Algorithm (DSA) was proposed in August 1991 by the U.S. National Institute of Standards and Technology (NIST). Digital signature authentication schemes provide secure communication with minimum computational cost for real time applications, such as electronic commerce, electronic voting, etc. The sender generates the signature of a given message using his secret key; the receiver then verifies the signature by using sender's public key. The ECDSA have a smaller key size, which leads to faster computation time and reduction in processing power, storage space and bandwidth. This makes the ECDSA ideal for constrained devices such as pagers, cellular phones and smart cards. The Elliptic-Curve Digital Signature Algorithm (ECDSA) is a Digital Signature Scheme based on ECC. ECDSA was first proposed in 1992 by Scott Vanstone in response of NIST (Nation Institute of Standards and Technology) request for public comments on their proposal for Digital Signature Schemes[1].

Digital Signature authenticated schemes, have the following properties.

1. **Confidentiality.** Secret information shared between sender and receiver; any outsider cannot read the information.
2. **Authentication.** The sender imprints his identity by means of the digital signature, which only the designated receiver can unravel and verify. An anonymous adversary cannot send a malicious message impersonating the genuine sender, because he does not have the necessary tools to generate the signature.
3. **Non-repudiation.** The signature firmly establishes the identity of the sender. The sender cannot deny having sent the message and the signature.

In this paper we discuss ECC in detail and ECDSA Implementation with different Text Message encryption methods and compared the results.

## II. Elliptic Curve Discrete Logarithm Problem

An elliptic curve E, [2] defined over a field K of characteristic $\neq 2$ or 3 is the set of solutions (x, y) $\in$ K' to the equation $y^2 = x^3 + ax + b$ (1)
a, b $\in$ K (where the cubic on the right has no multiple roots). Two nonnegative integers, a and b, less than p that satisfy:

$$4a^3 + 27b^2 \pmod p = 0 \qquad (2)$$

Then $E_p$ (a, b) denotes the elliptic group mod p whose elements (x, y) are pairs of nonnegative integers less than p satisfying:

$$y^2 = x^3 + ax + b \pmod p \qquad (3)$$

together with the point at infinity O.
The elliptic curve discrete logarithm problem(ECDLP) can be stated as follows. Fix a prime p and an elliptic curve.

$$Q = xP \qquad (4)$$

where xP represents the point P on elliptic curve added to itself x times. Then the elliptic curve discrete logarithm problem is to determine x given P and Q. It is relatively easy to calculate Q given x and P, but it is very hard to determine x given Q and P.

ECC is based on ECDLP. ECDH and ECDSA are cryptographic schemes based on ECC. The best known algorithm for solving ECDLP is Pollard-Rho algorithm which is fully exponential having a running time of $\sqrt{(\Pi * n / 2)}$.

## III. Elliptic Curve Cryptography

The Elliptic curve cryptosystems (ECC) were invented by Neal Koblitz [2] and Victor Miller[3] in 1985. They can be viewed as elliptic curve analogues of the older discrete logarithm (DL) cryptosystems in which the subgroup of $Z_p^*$ is replaced by the group of points on an elliptic curve over a finite field. The mathematical basis for the security of elliptic curve cryptosystems is the computational intractability of the elliptic curve discrete logarithm problem (ECDLP) [4].
ECC is a relative of discrete logarithm cryptography. An elliptic curve E over $Z_p$ as in Figure 1 is defined in the
Cartesian coordinate system by an equation of the form:

$$y^2 = x^3 + ax + b \qquad (5)$$

where a, b $\in Z_p$, and $4a^3 + 27b^2 \pmod p \neq 0 \pmod p$, together with a special point O, called the point at infinity. The set
$E(Z_p)$ consists of all points (x, y), x $\in Z_p$, y $\in Z_p$, which satisfy the defining equation, together with O.
Each value of a and b gives a different elliptic curve. The public key is a point on the curve and the private key is a random
number. The public key is obtained by multiplying the private key with a generator point G in the curve.
The definition of groups and finite fields, which are fundamental for the construction of elliptic curve cryptosystem are discussed in next subsections.

### 3.1. Groups

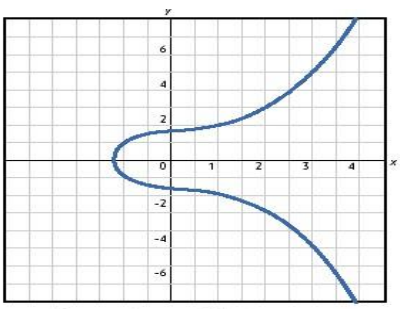


**Figure 1. An Elliptic Curve**

A group with an operation * is defined on pairs of elements of G. The operations satisfy the following properties:

- Closure: a * b $\in$G for all a, b $\in$G
- Associativity: a * (b * c) = (a * b) * c for all a, b, c $\in$G
- Existence of Identity: There exists an element e $\in$G, called the identity, such that e * a = a * e = a for all a $\in$G.
- Existence of Inverse: For each a $\in$G there is an element b$\in$G such that a * b = b * a = e. The element b is called the inverse of a.

Moreover, a group G is said to be abelian if a * b = b * a for all a, b $\in$G. The order of a group G is the number of elements in G.

### 3.2. Finite Field

A finite field consists of a finite set of elements together with two binary operations called addition and multiplication, which satisfy certain arithmetic properties. The order of a finite field is the number of elements in the field. There exists a finite field of order q if and only if q is a prime power. If q is a prime power, then there is essentially only one finite field of order q; this field is denoted by $F_q$. There are, however, many ways of representing the elements of $F_q$. Some representations may lead to more efficient implementations of the field arithmetic in hardware or in software. If q = $p^m$ where p is a prime and m is a positive integer, then p is called the characteristic of $F_q$ and m is called the extension degree of $F_q$.

#### 3.2.1. Prime Field $F_p$

Let p be a prime number. The finite field $F_p$ called a prime field, is comprised of the set of integers {0,1,2,….,p-1} with the following arithmetic operations:

- Addition: If a, b$\in F_p$ then a + b = r, where r is the remainder when a + b is divided by p and $0 \leq r \leq$ p-1 known as addition modulo p.
- Multiplication: If a, b $\in F_p$ then a.b = s, where s is the remainder when a.b is divided by p and $0 \leq s \leq$ p-1 known as multiplication modulo p.
- Inversion: If a is non-zero element in $F_p$, the inverse of modulo a modulo p, denoted by $a^{-1}$, is the unique integer c$\in F_p$ for which a.c = 1.

#### 3.2.2. Binary Field $F_2^m$

The field $F_2^m$, called a characteristic two finite field or a binary finite field, can be viewed as a vector space of dimension m over the field $F_2$ which consists of the two elements 0 and1. That is, there exist m elements $\alpha_0, \alpha_1, \ldots, \alpha_{m-1}$ in $F_2^m$ such that each element α can be uniquely written in the form:

$\alpha = a_0\ \alpha_0 + a_1\ \alpha_1 + \ldots\ldots\ldots + a_{m-1}\ \alpha_{m-1}$, where $a_i \in \{0,1\}$

Such a set $\{\alpha_0, \alpha_1, \ldots, \alpha_{m-1}\}$ is called a basis of $F_2^m$ over $F_2$. Given such a basis, a field element α can be represented as the bit string ($a_0$ + $a_1$ ……….+$a_{m-1}$) Addition of field elements is performed by bitwise XOR-ing the vector representations. The multiplication rule depends on the basis selected. ANSI X9.62 permits two kinds of bases: polynomial bases and normal bases.

#### 3.2.3. Domain Parameters

The domain parameters for ECDSA consist of a suitably chosen elliptic curve E defined over a finite field $F_q$ of characteristic p, and a base point G $\in$E(Fq). Domain parameters may either be shared by a group of entities, or specific to a single user. To summarize, domain parameters are comprised of:

1. A field size q, where either q = p, an odd prime, or q = $2^m$
2. An indication FR (field representation) of the representation used for the elements of $F_q$
3. (optional) a bit string seed E of length at least 160 bits
4. Two field elements a and b in $F_q$ which define the equation of the elliptic curve E over $F_q$' (i.e., $y^2 = x^3 + ax + b$ in the case p > 3, and $y^2 + xy = x^3 + ax + b$ in the case p = 2)
5. Two field elements $x_G$ and $y_G$ in $F_q$ which define a finite point G = ($x_G$, $y_G$) of prime order in E($F_q$)
6. The order of the point G, with $n>2^{160}$ and $n > 4\sqrt{q}$ and
7. The cofactor h = #E($F_q$)/n

### 3.3. Elliptic Curve Operations over Finite Fields[8]

The main operation is Point multiplication is achieved by two basic elliptic curve operations.
i. Point addition, adding two points P and Q to obtain another point R i.e. R= P + Q. ii. Point doubling, adding a point P to itself to obtain another point R i.e. R = 2P.

#### 3.3.1. Point Addition

Point addition is the addition of two points P and Q on an elliptic curve to obtain another point R on the same elliptic curve.
Consider two points P and Q on an elliptic curve as shown in Figure 2. If P ≠ -Q then a line drawn through the points P and Q will intersect the elliptic curve at exactly one more point –R. The reflection of the point –R with respect to x-axis gives the point R, which is the result of addition of points P and Q. Thus on an elliptic curve R = P + Q. If Q = -P the line through this point intersect at a point at infinity O. Hence P + (-P) = O. A negative of a point is the reflection of that point with respect to x-axis.

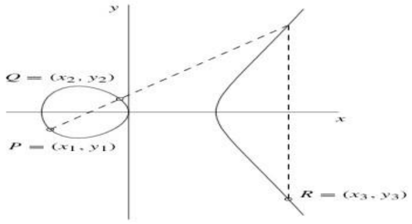


**Figure 2: Point Addition**

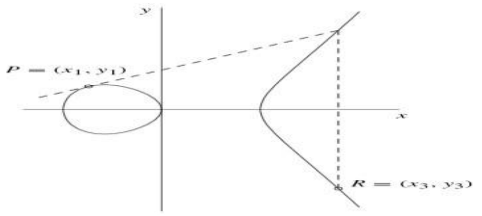


**Figure 3: Point Doubling**

#### 3.3.2. Point Doubling

Point doubling is the addition of a point P on the elliptic curve to itself to obtain another point R on the same elliptic curve
To double a point J to get L, i.e. to find R = 2P, consider a point P on an elliptic curve as shown in Figure 3. If y coordinate of the point P is not zero then the tangent line at P will intersect the elliptic curve at exactly one more point –R. The reflection of the point –R with respect to x-axis gives the point R, which is the result of doubling the point P, i.e., R = 2P. If y coordinate of the point P is zero then the tangent at this point intersects at a point at infinity O. Hence 2P = O when $y_j = 0$. Figure 3 shows point doubling.

#### 3.3.3. Algebraic Formulae over $F_p$

Let p be a prime in $F_p$ and a, b∈ FP such that $4a^3 + 27b^2 \neq 0 \bmod p$ in $F_p$, then an elliptic curve E ($F_p$) is defined as E ($F_p$):= { p( x, y) , x, y ∈ $F_p$ } Such that $y^2 = x^3 + ax + b \bmod p$ together with a point O, called the point at infinity. Below is the definition of addition of
points P and Q on the elliptic curve E ($F_p$). Let P($x_1$, $y_1$) and Q($x_2$, $y_2$) then

$$R = P+Q = \begin{cases} \text{If } x_1 = x_2 \text{ and } y_2 = -y_1 \\ Q = Q+P \quad \text{If } P = O \\ (x_3, y_3) \quad \text{otherwise} \end{cases}$$

Where $x_3$ =

$$\begin{cases} \lambda^2 - x_1 - x_2 & \text{If } P \neq \pm Q \text{ (Point Addition)} \\ \lambda^2 - 2x_1 & \text{If } P = Q \text{ (Point Doubling)} \end{cases}$$

$y_3 = \lambda(x_1 - x_3) - y_1$, and

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{If } P \neq \pm Q \text{ (Point Addition)} \\ \dfrac{3x_1^2 + a}{2y_1} & \text{If } P = Q \text{ (Point Doubling)} \end{cases}$$

The point p(x, -y) is said to be the negation of p(x, y).

### 3.3.4. Algebraic Formulae over $F_2^m$

Denote the (non-super singular) elliptic curve over $F_2^m$ by E ($F_2^m$). If a, b $\in F_2^m$ such that $b \neq 0$ then E ($F_2^m$) = {p(x, y), x, y $\in F_2^m$ } such that $y^2 + xy = x^3 + ax^2 + b \in F_P$ together with a point O, called the point at infinity. The addition of points on E ($F_2^m$) is given as follows: Let P($x_1$, $y_1$) and Q($x_2$, $y_2$) be points on the elliptic curve E($F_2^m$), then

$$R = P+Q = \begin{cases} O & \text{If } x_1 = x_2 \text{ and } y_2 = -y_1 \\ Q = Q+P & \text{If } P = O \\ (x_3, y_3) & \text{otherwise} \end{cases}$$

Where

$$x_3 = \begin{cases} \lambda^2 + \lambda + x_2 + x_1 + a & \text{If } P \neq \pm Q \text{ (Point Addition)} \\ \lambda^2 + \lambda + a & \text{If } P = Q \text{ (Point Doubling)} \end{cases}$$

$y_3 = \lambda ( x_1 + x_3 ) + x_3 + y_1$ and

$$\lambda = \begin{cases} \dfrac{y_2 + y_1}{x_2 + x_1} & \text{If } P \neq \pm Q \text{ (Point Addition)} \\ x_1 + \dfrac{y_1}{x_1} & \text{If } P = Q \text{ (Point Doubling)} \end{cases}$$

## IV. Implementation

This paper presents VTM Encryption, VTM decryption [5], ECDSA key generation, signature generation and signature verification algorithms [8] and ECDSA was implemented over Elliptic Curve (EC) P-192 and P-256 using Text Message Encryption methods which are VTM [5], FTM[5] and TBM [6] encryption methods and compared their performance.

**Algorithm-1**

VTM Encryption Algorithm[5]

**NOTATION:** TM - Text message

M - Message units VS - variable size IV - Initial Vector

k - Auxiliary base parameter

XRM - XORed message

Block – a word with followed space

**INPUT:** sextuple T = (p, a, b, G, n, h), Digest Message

**OUTPUT:** Encrypted Message

**Begin**

n = wordCount(DM)

**for** i = 1 to n **do**

XRM = IV ⊕ Block[i] M = ASCII(XRM)

**for** j = 0 to k-1 **do**

let $x_j$ = M * K + j mod p

3

**if** $z_j = x_j$

**break**

**end if end for**

**if** j < k **then**

+ $x_j$ + b has a square root mod p **then**

compute $y_j$ a square root of $z_j$ mod p map M to $(x_j, y_j)$
**else**
output "unsuccessful in attempt to map M to an EC point"
**end if**
Cm[i] = { kG ,Pm + $kP_B$} IV = XRM
**end for**
**End**

**Algorithm-2**

VTM Decryption Algorithm[5]

**INPUT:** sextuple T = (p, a, b, G, n, h), Encrypted Message

**OUTPUT:** Decrypted/Plain Digest Message

**Begin**
**for** i = 1 to n **do** //where n is number of cipher texts
Pm(x, y) = Pm + K($n_B$G) **-** $n_B$(kG) // $n_B$ receivers private key
M = x/k
Dm = Text(M) // M is decimal value of base 256 format
TM[i] = Dm ⊕ IV IV = Dm

TM = TM || TM[i]
**end for**
**End**

**Algorithm-3**

ECDSA Key pair generation Algorithm[8]

**INPUT:** Domain parameters D= (q, FR, a, b, G, n, h).

**OUTPUT:** Public key Q, private key d.

Select d∈ [1, … , n-1] Compute Q = dG Return (Q, d)

**Algorithm-4**

ECDSA Signature Generation Algorithm[8]

**INPUT:** Domain parameters D= (q, FR, a, b, G, n, h) , private key d, Encrypted message m'.

**OUTPUT:** Signature (r,s)

**begin repeat**
k = Random[1, … , n-1] // select random value
r = x-coord([k]G) mod n e = H(m')
s = $k^{-1}$(e+dr) mod n
**until** r ≠ 0 and s ≠ 0
**return** (r,s).
**end**

**Algorithm-5**

ECDSA Signature Verification Algorithm[8]

**INPUT:** Domain parameters D= (q, FR, a, b, G, n, h) , public key Q, Encrypted Message m', Signature (r, s).
**OUTPUT:** Acceptance or rejection of the signature.
**begin**
**if** r, s ∉ [1,…, n] **then**
Return ("Reject the signature")

**end if**
e = H(m')
w = $s^{-1}$ mod n $u_1$ = ew mod n $u_2$ = rw mod n

x = $u_1G + u_2Q$
**if** x = ∞ **then**
Return (“Reject the signature”)
**end if**
**v** = x-coord( X ) mod n
**if** v = r **then**
Return (“Accept the signature”)
**else**
Return (“Reject the signature”)
**end if**
**end.** Elliptic Curve based Signature Generation & Signature Verification processes are described below and the same is represented in graphical format in figure 4 and figure 5.

Signature Generation steps:
1. Digest the plain message using SHA algorithm.
2. Encrypt the message using EC Encryption algorithm which is VTM/FTM/TBM
3. Compute signature for Encrypted message using Algarithm-4
3. Send the digitally signed message

Signature Verification Steps:
1. Verify Signature using Algarithm-5.
2. If verification fails then reject the signature
3. If verification success, then decrypt the message using respective EC Decryption Algorithm.

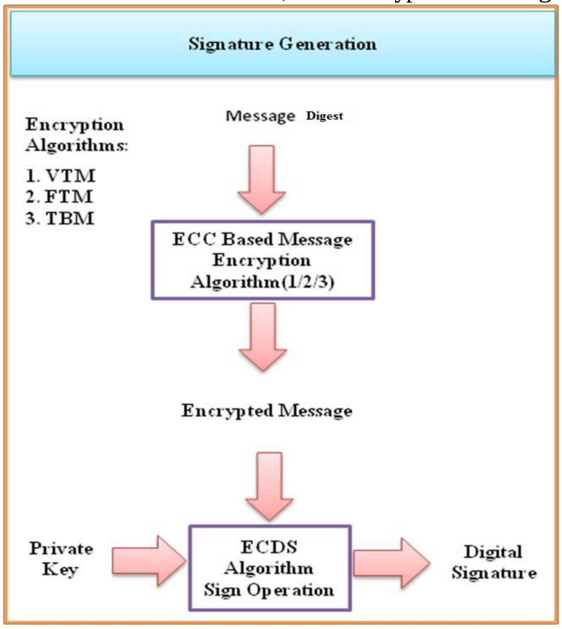


**Figure 4: Signature Generation Process**

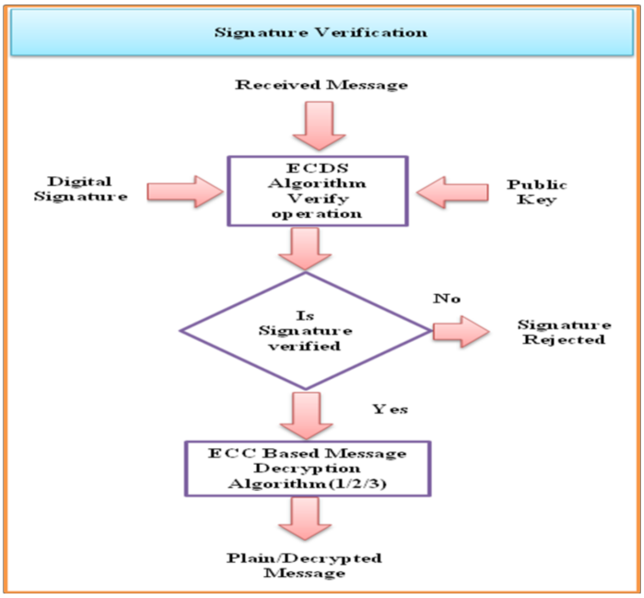


**Figure 5: Signature Verification Process**

## V. Results and Discussion

In this section represents implementation results of ECDSA using VTM encryption over EC P-192 and P-256.

### 5.1. Results over Elliptic Curve P-192

Message m = " hello this is a raajasekhar from kurnool so plan to go us in the next year for working in DW Practice LL atlanta georgia "
Private key = 2055107281
Public Key = (5841942716391479201550342297351085963270983519924994377602, 5584890377300947026793868981513336619407548239394095574193)
This message encrypted and follows Signature Generation and Verification as mentioned below. Encrypted message hash value H(E(m)) = -2682108996977278156968408606235438945161064554

- ECDSA SIGNATURE as follows: Select k= 1583021364

Compute kG = (37921946278159604401180029145945511663128641788889626308 82,
289119065962065605999071802266214672856485360554016800 1982)

Compute kG = (3792194627815960440118002914594551166312864178888962630882,
2891190659620656059990718022662146728564853605540168001982)
r = 62742A904369649DB4FD7CAD870EA7E7D2058DD5
Compute s = $k^{-1}$ (e + dr) mod n = 3411184681610252308390502359065554562708605093739075483483
Signature for the message m is (r, s).

- ECDSA VERIFICATION as follows:

Compute w = 5777480145803669741573423688926176979417082505271032360268
Compute $u_1$ = 4666422527249034100042022946337090008510597277184111303696 $u_2$ = 4455907927429886473277204474990236853124877171335661271649
$u_1G$ = (3929708989969467697197486716672122446942315632094831043367,
4537003456571103380284504813721792096119198047543959491671)
$u_2Q$ = (1277661715800205348067420766016806475954133626929696383370,
4380808460387567649107054289732585886848088206125448742447)
v = 62742A904369649DB4FD7CAD870EA7E7D2058DD5
We obtain v = r, that is accept the signature.

### 5.2. Results over Elliptic Curve P-256

Message m= " The Elliptic Curve Digital Signature Algorithm Validation System (ECDSAVS) specifies the procedures involved in validating implementations of the Elliptic Curve Digital Signature Algorithm(ECDSA) as approved in FIPS 186-2, Digital Signature Standard (DSS)[1] and specified in ANSIX9.62-1998, Public Key Cryptography for Financial Services Industry: The Elliptic CurveDigital Signature Algorithm (ECDSA)[2]. The ECDSAVS is designed to perform automated testing on Implementations Under Test (IUTs). This document provides the basic design and configuration of the ECDSAVS. "
Private Key = 978425864
Public Key = (118910487909274429022743485742135581553673510998540082125096949934 59447093822,
136698797209684711142721957596171372481001364004993589753744001635050 99163986) This message encrypted and follows Signature Generation and Verification as mentioned below.
Encrypted message hash value H(E(m)) = 5377030903796497704021953970510623230690 92491846

- ECDSA SIGNATURE as follows:

Select k= 115792089210356248762697446949407573529996955224135760342422259061068383502243
Compute
KG = (86500881224166483227925267313354237293018428812409245047778807509807358555053,
39579053610346434470532506438011786967057506613223689314593851851982117599776)
r = 86500881224166483227925267313354237293018428812409245047778807509807358555053
Compute s = $k^{-1}$ (e + dr) mod n
= 104389700715501732796614779737855463749375844486540618622018054702970561091708
Signature for the message m is (r, s).

- ECDSA VERIFICATION as follows:

Compute w = 106506396977556145535418054052339447393078832993181450002668470251312371474276
Compute $u_1$ =43824495211803284954034352427133274304161118431427286644319226927046995 29209
$u_2$=57692616982311160984176366728847647733800539362706147029132815066162592219439
$u_1G$ = (101474627893392564150949213703200203728873111984 8 92002825714765996844262058436,
60937423109159230990348336949980 80 456436196569064621167172651499915 1554795408)
$u_2Q$ = (109322103145683055628956971282445177307378355734712278598030249871906512163766,
42753639382524136274231334284305572212602843186842236043136827079395299552547)
v = 86500881224166483227925267313354237293018428812409245047778807509807358555053
We obtain v = r, that is accept the signature.
In the same way we have used TBM with plain message and encrypted for Signature generation and signature verification. ECDSA using Variable Size Text Message Encryption is better in performance aspect when compare with the other two methods and the results comparison is presented graphically in the next section.

## VI. Comparison Of ECDSA Using Various Text Based Cryptosystems

We compare the results of ECDSA using Text Based Message with plain text(TBM) Encryption[6] and Text Based Message with message digest[6]. Figure 6 and Figure 7 presents total time taken for Signature Generation and Signature Verification when we use different text based encryption methods in ECDSA implementation. From Figure 6 and Figure 7, performance of ECDSA using Text Based Message with message digest Encryption is better when compare with ECDSA using TBM with plain text. The reason is TBM based ECDSA used message digest compare with other one method. Performance of ECDSA is inversely proportional to key size, and security of the system depends on key size.

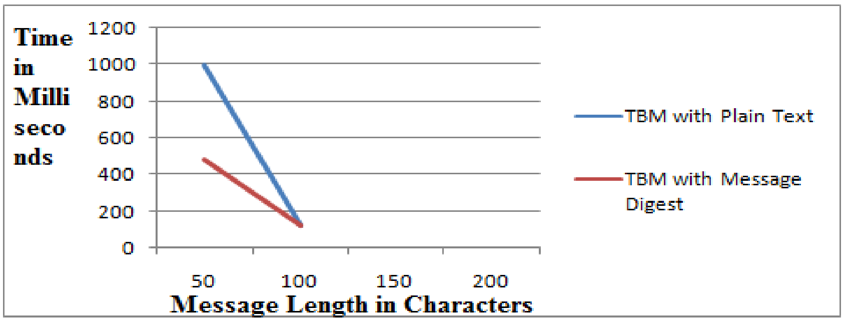


Figure 6: Performance comparison of various ECDSA methods for over EC P-192

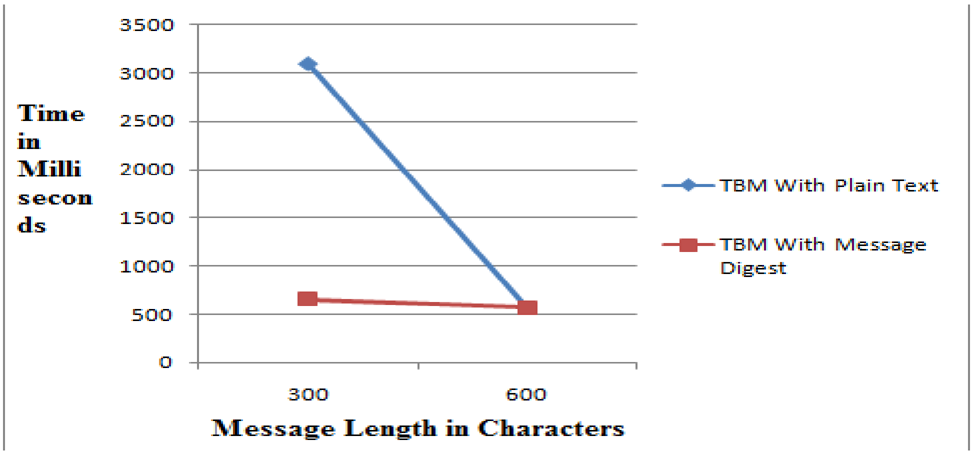


Figure 7: Performance comparision of various ECDSA methods for over EC P-256

## VII. Conclusion

In this paper we have implemented ECDSA for various domain parameters, after observing the results when the key size increases then complexity increases and performance decreased. After comparing TBM with plain message and message digest based ECDSA methods, ECDSA using Variable Text Message Encryption with message digest is better when comparing with plain text Encryption used ECDSA. The main reason is, the speed of scalar multiplication which plays an important role in the efficiency of whole system [7]. In VTM based ECDSA method, number of scalar multiplications are reduced, so this method is efficient when compared with FTM and TBM based methods.

## References

[1] Navneet Randhawa, Lolita Singh, A Systematic Way to Provide Security for Digital Signature Using Elliptic Curve Cryptography, IJCST Vol.2, Issue 3, Sep-2011, 185-188

[2] Koblitz, N., 1987. Elliptic curve cryptosystems. Mathematics of Computation 48, 203-209. [3] Miller, V., 1985. Use of elliptic curves in cryptography. CRYPTO 85.

[4] Certicom ECC Challenge. 2009. Certicom Research

[5] Jayabhaskar Muthukuru, Bachala Sathyanarayana, Fixed and Variable Size Text Based Message Mapping Techniques Using ECC, GJCST Vol.12, Issue 3, Feb-2012, 25-30.

[6] S. Maria Celestin Vigila , K. Muneeswaran “Implementation of Text based Cryptosystem using Elliptic Curve Cryptography”, IEEE Sep-2009, pp. 82-85.

[7] Harsandeep Brar , Rajpreet Kaur, “Design and Implementation of Block Method for Computing NAF” IJCA, Volume 20– No.1, April 2011, pp. 37-41.

[8] Hankerson, D., Menezes, A., Vanstone, S., Guide to Elliptic Curve Cryptography (Springer, 2004).