FAIL-STOP DESIGNATED RECIPIENT SIGNATURE SCHEME BASED ON ELLIPTIC CURVE DISCRETE LOGARITHM PROBLEM

Nedal Tahat

Department of Mathematics
Faculty of Sciences
The Hashemite University
Zarqa 13133
Jordan
e-mail: nedal@hu.edu.jo

Abstract. This paper describes a new digital signature called fail-stop designated recipient signature scheme based on elliptic curve discrete logarithm (ECDLP). The scheme allows a signer and an intended recipient to co-operatively provide a proof of forgery if an attacker can successfully forge a signature on a message m. The scheme also provides that the intended recipient is the only entity to verify the resulting signature and capable to prove the validity of signature to any third party. With this property, we show that our new signature scheme is the best alternative to solve certain problems concerning the protection of confidential documents especially those which are personally sensitive (to the owner) and is also applicable for group of recipients to verify a signature jointly (shared verification) in a group oriented environment. The scheme utilizes the inherent advantage of elliptic curve cryptosystem in terms of smaller key size and lower computational overhead to its counterpart public cryptosystems such as RSA and AlGamal. Compared with the Ismail and Yahya scheme [1], we show that our scheme is more efficient in terms of both computation and communication complexity.

Keywords: Cryptography, Digital signature, Elliptic curve discrete logarithm problem, Fail-stop designated recipient signature and zero knowledge Protocol.

1. Introduction

In all classical digital signatures if a forger successfully obtains a forge signature that pass the verification procedure then the scheme is completely insecure and the signer would be blamed. This is the main disadvantage of the digital signature scheme. However, fail-stop signature completely avoids this type of attack. If an attacker successfully forger has happened. This concept was first introduced by Waidner and Pfitzmann [13] and then was formally defined in pfitzmann and Waidner (1990). Theoretically, fail-stop signature are known to exist if claw-free pairs of permutations (not necessarily with trapdoor) exist, see [2] and [19] for description and Pfitzmann and Waidner [13] for proof. For complete definition of claw-free permutations, see Goldwasser [3]. In particular, [2] and [5] show that fail-stop signature only exist if factoring large integer or discrete logarithm problem is hard. The first construction of fail-stop signature [13] uses a one-time signature scheme and requires message to be signed bit although the tree-authentication [12] is used. This general construction is not efficient. There is an efficient construction for fail-stop signature that can be used to protect clients unconditionally secure in online payment system [19]. However in this scheme all signatures by one client must have the same recipient like the bank in a payment system. Furthermore, signing is a 3-round protocol between the signer and the recipient. Pftitzmann [18] also presented an efficient scheme with single recipient. First fail-stop signature scheme based on factorization and discrete logarithm problems have been proposed respectively in Susilo et. al [23] and Heyst and Pedersen [4]. The former uses modulus factor as a proof of forgery whereas the latter uses a private key of the system as a proof of forgery. In Pedersen and Pfitzmann [16], a formal definition of fail-stop signature is given and a general construction using bundling homomorphism is proposed. See also Pfitzmann [17]. To provide a proof of forgery, they show that under bundling homomorphism, two signatures (respectively generated by signer and forger) are shown to collide. In 2007, Ismail and Yahya[1] fail-stop recipient designated signature scheme based on RSA have been proposed. The elliptic curve cryptosystems ECC was initiated by Koblitz [6] and Miller [14], where the security was established on the discrete logarithm problem over the points on an elliptic curve, called ECDLP. The basic operations are the execution of integer points on the elliptic curve over finite fields, including addition and multiplication. The operations associated with the ECC are more efficient than those associated with other cryptosystems, like the RSA and the DSA security solutions. Owing to the fact that the ECC has a smaller key size and faster computation, for this reason, in this paper we had the motivation to design a new type from fail-stop designated recipient signature based on ECDLP.

The paper is organized as follows.

In the next section, we present the basic concept and definition of fail-stop designated recipient signature (FDRS) and introduce the notations that are used throughout the rest of the paper.

In Section 3, we propose our new scheme.

In Sections 4 and 5, we analyze security and efficiency performance of the proposed scheme.

Next, we present a small example of our scheme and this can be found in Section 6.

Finally, Section 7 concludes the paper.

2. Preliminaries

2.1. The model of fail-stop designated recipient signature

FDRS consists of five algorithms. The algorithms are as in original fail-stop signature, but we put some modifications to suit our desired applications.

Definition 2.1 A fail-stop designated recipient signature scheme consists of five algorithms (GEN, SIGN, VER, PVER, PTEST).

- GEN, SIGN and VER are respectively for generating keys, signing messages and verifying signatures.
- PVER for recipient to prove validation of signature to a third party without revealing any secret information
- PTEST for both signer and recipient to provide a proof of forgery if an adversary successfully forges a signature.

A secure FDRS must satisfy the following properties:

1. If the signer signs a message, then the recipient can verify the signature and accept it as genuine
2. A polynomial bounded (limited) forger cannot create forged signatures that successfully pass the verification test.
3. When a forger with unlimited computational power succeeds in forging a signature that passes the VER, the signer and recipient are able to construct a proof of forgery and convince an interested third party that a forgery has occurred.

A polynomials bounded (limited) signer or recipient cannot create a signature that he can later prove to be a forgery.

To achieve the above properties, for each public key, there must exist many matching secret keys. In other words, there must exist secret keys, SK, a set consisting of secret keys and each of which fits with the signer's public key, PK. Different secret keys from SK must create different signatures on the same message, m. The real signer knows only one of the secret keys and thus can only construct one possible signature on a given message. An enemy with unlimited computational power knows all the secret keys in SK, thus can generate all the signatures but he is unable to determine which one will be used by the signer. If the enemy presents a forged signature and claiming that the signature is signed by a signer, the signer then is able to provide a proof of forgery by generating his own signature on the message and use this signature with the one presented by enemy to break the underlying assumption of the scheme.

To show security of FDRS, it suffices to prove the following properties:

- There exists a probabilistic polynomial time algorithm PVER that takes a pair of secret and public key, a message and a forged signature for that message, and outputs a proof of forgery.
- An enemy with unlimited computing power, who knows the public key of the signer and his or her signature on a message, cannot find the secret key of the signer. Thus, he or she would not be able to construct signer's signature on a new message
- A polynomially bounded signer cannot construct valid signature on a message, and later prove that it is a forgery.

Generally, FDRS can be categorized into two ways: (1) FDRS with a trusted dealer (TD) who is trusted by all other entities and the one who is responsible in initializing the system and (2) FDRS without TD, and in this case the role of TD is given to a recipient. Our proposed scheme is based on the former.

2.2. Elliptic curve

Elliptic curve cryptosystem have attracted much attention in recent years because of the relatively small size of keys they require. An elliptic over $GF(p)$ is the set of points $(x, y) \in GF(p)$ satisfying the equation

$$y^2 = x^3 + ax + b$$

together with a special element denoted $\mathcal{O}$ and called the point at infinity. Addition operation on the points of an elliptic curve can be defined that makes it into an abelian group.

For more precise definition operation, we refer the reader to [11].

Let $E_p(a, b)$ denotes an elliptic curve of the form

$$y^2 = (x^3 + ax + b)(\text{mod} p),$$

where p is a prime number, $x^3 + ax + b(\text{mod} p)$ does not have multiple roots, and

$$4a^3 + 27b^2 \neq 0(\text{mod} p)$$

$|E_p(a, b)|$ denotes the order of $E_p(a, b)$ and can be calculated in polynomial time using algorithm such as Schoof's algorithm [8, 20,24], and combination of Schoof's algorithm with Shank's baby-step giant-step algorithm [11].

Definition 2.2 An elliptic curve discrete logarithm problem (ECDL) is defined as follows: Let $\alpha \in E_p(a, b)$ be point of order q, and let $\beta = d\alpha$. Given α and β, determine the unique integer d, where $0 < d \leq q$

3. The proposed scheme

We now present our fail-stop designated recipient signature scheme based on the elliptic curve discrete logarithm problems.

3.1. The system setup

The fail-stop designated recipient signature scheme requires TD to choose and generate the system parameters for the scheme. TD is implemented according to the following steps:

- TD chooses an elliptic curve such that $q = |E_p(a, b)|$ is also prime.
- TD randomly chooses a point $\alpha \in E_p(a, b)$ and a number $d \in GF(q)$.
- calculates $\beta = d\alpha$ over and discards d.

Finally he publishes $(E_p(a, b), q, \alpha, \beta)$.

A simple algorithm to find such a curve is to randomly select a curve, find its order, discard the curve if the order is non-prime and repeat the process. It is conjectured that this type of curve can be obtained in $\mathcal{O}(1/log\ p)$ [7].

The signer then identifies the intended recipient who wishes to accept a resulting signature. The signer and intended recipient then agree on a common secret key $\lambda \in GF(q)$. Next the signer passes securely the integer β to the intended recipient who then does the following:

- Selects at random private key $v \in GF(q)$ and computes and securely sends $\gamma = v\beta$ over $E_p(a, b)$ to the signer.

The signer then proceeds by selecting secret keys and computing public keys as shown below:

- Chooses four secret keys $k_1, k_2, k_3, k_4 \in GF(q)$
- Computes the following public keys that are related to the intended recipient:

$$\begin{aligned}\beta_1 &= (k_4\alpha + k_3\gamma)\ (over\ E_p(a, b))\\ \alpha_1 &= (k_3\alpha + k_1\beta_1)\ (over\ E_p(a, b))\\ \alpha_2 &= (k_4\alpha + k_2\beta_1)\ (over\ E_p(a, b))\end{aligned}$$

The four secret keys chosen by signer must be used once (choose different secret keys for different messages). This is because if two different messages are signed using the same four secret keys then these secret keys will be obtained easily.

3.2. Signing and verifying

To generate the signature for a message $m \in GF(q)$, the signer computes

$$\begin{aligned}y_1 &= (k_1m + k_2\lambda)(\text{mod}q)\\ y_2 &= (k_3m + k_4\lambda)(\text{mod}q)\end{aligned}$$

and publishes (y_1, y_2) as his signature on m.

The recipient accepts the signature as genuine if and only if the following check is correct:

$$y_2\alpha + y_1\beta_1 = (m\alpha_1 + \lambda\alpha_2)\ (over\ E_p(a,b)) \tag{3.1}$$

Note that only the intended recipient, who knows λ can verify the signature. In some cases, the recipient needs to prove the validity of the signature to any third party. To do this the recipient and the signer must cooperate where the signer computes and sends recipient $w_1 = k_1\beta_1$ and $w_2 = k_3\alpha$ secretly. The recipient then sends third party $w_3 = \lambda\alpha_2$ who next confirms that

$$y_2\alpha + y_1\beta_1 - m\alpha_1 = w_3$$

Finally, the recipient proves that $m(w_1 + w_2) = m_1$ in a Zero-knowledge technique for example using the protocol based on the elliptic curve discrete logarithm [10].

Now, assume that Adv successfully obtains a forged signature $(\bar{y}_1, \bar{y}_2)$ that passes VER, the signer and the recipient now can jointly prove that a forgery has happened by running the following procedure:

3.3. Algorithm proof of forgery (PTEST)

- Construct his own signature on message m as (y_1, y_2)
- Signer then calculates $(y_2 - \bar{y}_2)$ and $(\bar{y}_1 - y_1)$
- Signer and recipient jointly compute $d = \dfrac{(y_2 - \bar{y}_2) - (\bar{y}_1 - y_1)k_4}{(\bar{y}_1 - y_1)k_3 v}$ (mod$q$) and use this as the proof of forgery.

Theorem 3.1 *If GEN and SIGN are run smoothly the validation in VER is correct.*

Proof. Let a signature on a message m is given by (y_1, y_2). Then it is easy to show that the following congruence holds:

$$\begin{aligned} y_2\alpha + y_1\beta_1 &= (k_3m + k_4\lambda)\alpha + (k_1m + k_{2\lambda})\beta_1 \\ &= (\alpha k_3 m + \alpha k_4\lambda + \beta_1 k_1 m + \beta_1 k_2\lambda) \\ &= (\alpha k_3 + \beta_1 k_1)m + (\alpha k_4 + \beta_1 k_2)\lambda \\ &= m\alpha_1 + \lambda\alpha_2 \end{aligned}$$ ■

4. Security and efficiency performance

4.1. Security

An enemy with unlimited computational power has a non-negligible probability to forge a signature and if the enemy presents the forge signature, we can provide a proof that a forgery has happened. In our scheme, t provides such proof; we need collaboration between a signer and a designated recipient. To show that our scheme is secure, we have to prove the following lemmas and theorems as suggested in Section 2. We begin by proving the following lemma.

Lemma 4.1 *There are q^2 equally likely secret keys that mach with the signer related public key.*

Proof. The public key of scheme $(\alpha_1, \alpha_2, \beta_1)$ gives the following three equations:

$$\begin{aligned}\beta_1 &= k_4\alpha + k_3\gamma(over E_p(a,b))\\ \alpha_1 &= k_3\alpha + k_1\beta_1(over E_p(a,b))\\ \alpha_2 &= k_3\alpha + k_2\beta 1(over E_p(a,b))\end{aligned}$$

Since $\gamma = v\beta = vd\alpha$, we have:

$$\beta_1 = k_4\alpha + k_3vd\alpha = (k_4 + k_3vd)\alpha.$$

Solving the ECDLP problem we have:

$$c = k_4 + k_3vd, \quad \text{where } c \in GF(q).$$

We also have:

$$\alpha_1 = k_3\alpha + k_1(k_4 + k_3vd)\alpha = (k_3 + k_1c)\alpha$$

and

$$\alpha_2 = k_4\alpha + k_2(k_4 + k_3vd)\alpha = (k_4 + k_2c)\alpha.$$

Solving the ECDLP problem, we have:

$$\begin{aligned}k_3 + k_1c &= c_1 \pmod q\\ k_4 + k_2c &= c_2 \pmod q\end{aligned}$$

where $c_1, c_2 \in GF(q)$. Equivalently,

$$\begin{pmatrix} c & 0 & 1 & 0\\ 0 & c & 0 & 1\end{pmatrix}\begin{pmatrix}k_1\\k_2\\k_3\\k_4\end{pmatrix} = \begin{pmatrix}c_1\\c_2\end{pmatrix}(\text{mod}q)$$

This is a set of 2 linear equations in 4 unknowns where the rank of the coefficient matrix is equal to 2. Hence, there are q^2 solutions corresponding to assigning arbitrary values (mod$q$) to k_3 and k_4 (q^2 possibilities), and calculating the values of k_1 and k_2.

Theorem 4.2 *If the signer receives a forged signature $(\bar{y}_1, \bar{y}_2)$ not equal to his valid signature (y_1, y_2) on a message m and the forge signature passes the verification test then he is able to solve ECDL problem.*

Proof. Since both signatures (y_1, y_2) and $(\bar{y}_1, \bar{y}_2)$ pass the verification test, then, according to (3.1), we must have that:

$$\begin{aligned}y_2\alpha + y_1\beta_1 &= \bar{y}_2\alpha + \bar{y}_1\beta_1(\text{over } E_p(a,b))\\ y_2\alpha - \bar{y}_2\alpha &= \bar{y}_1\beta_1 - y_1\beta_1(\text{over } E_p(a,b))\\ (y_2 - \bar{y}_2)\alpha &= (\bar{y}_1 - y_1)\beta_1(\text{over } E_p(a,b))\end{aligned}$$

$$
\begin{array}{lrcl}
(4.1) & (y_2-\bar{y}_2)\alpha & = & (\bar{y}_1-y_1)(k_4+k_3vd)\alpha\,(\text{over}\,E_p(a,b)) \\
(4.2) & (y_2-\bar{y}_2) & = & (\bar{y}_1-y_1)k_4+(\bar{y}_1-y_1)k_3vd\,(\text{mod}q) \\
(4.3) & (y_2-\bar{y}_2)-(\bar{y}_1-y_1)k_4 & = & (\bar{y}_1-y_1)k_3vd\,(\text{mod}q) \\
(4.4) & d & = & \dfrac{(y_2-\bar{y}_2)-(\bar{y}_1-y_1)k_4}{(\bar{y}_1-y_1)k_3v}
\end{array}
$$

The correctness of driving equation (4.2) from (4.1) is ensured by the following lemmas.

Lemma 4.3 [11], [22], [8] *Any elliptic curve $E_p(a,b)$ where $q=|E_p(a,b)|$ is also prime, forms a cyclic group, which is isomorphic to $GF(q)$. Thus, any point other than the point at infinity is a generator of $E_p(a,b)$.*

Lemma 4.4 *If there is an equation of the form, $c\mu = a\mu + b\mu$ over a curve $E_p(a,b)$, where $a,b,c \in GF(q)$, $\mu \in E_p(a,b)$, and $q=|E_p(a,b)|$ is prime, then we have*

$$h=(a+b)(mod\, q)$$

.

Proof. According to Lemma 4.2, every point on $E_p(a,b)$, including μ, is a generator of the group and has order q, where $q=|E_p(a,b)|$.

Theorem 4.5 *Knowing the public key together with the signature for a message m, an enemy unlimited computational power can calculate q possible secret keys that could have been used for signing the message.*

Proof. With the public key $(\alpha_1,\alpha_2,\beta_1)$ and the signature (y_1,y_2) on m, an unlimited computational power enemy can solve the ECDL problem and rewrite these equations as follows:

$$
\begin{array}{rcl}
c_1 & = & (k_3+k_1c)(mod\, q) \\
c_2 & = & (k_4+k_2c)(mod\, q) \\
\bar{y}_1 & = & (k_1m+k_2)(mod\, q) \\
\bar{y}_2 & = & (k_3m+k_4)(mod\, q)
\end{array}
$$

where $c=(k_4+k_3vd), c_1, c_2 \in GF(q)$ and the last two equations are an acceptable signature on m. These equations next can be formed as

$$
\begin{pmatrix} c & 0 & 1 & 0 \\ 0 & c & 0 & 1 \\ m & 1 & 0 & 0 \\ 0 & 0 & m & 1 \end{pmatrix}
\begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \end{pmatrix}
=
\begin{pmatrix} c_1 \\ c_2 \\ \bar{y}_1 \\ \bar{y}_2 \end{pmatrix}
(\text{mod}\, q)
$$

It is easy to see that this matrix has rank 3 (this is true because $cr_3 - r_2 - mr_1 + r_4 = 0$, where r_i is the i^{th} row of the matrix and noting that the submatrix consisting of the first 3 columns has 3 independent rows), and so there are exactly q solutions to this equation.

Theorem 4.6 *The signer can prove a forgery with probability equal* $\frac{q-1}{q}\cdot$

Proof. Given a forged signature that passes the verification test, the presumed signer can generate a different signature which passes the verification test with probability $\frac{q-1}{q}$, where q is the number of possible signatures on a message (this can be seen from previous Theorem 4.4).

Corollary 4.7 *A computationally bounded signer cannot make signatures which he can later prove to be forgeries.*

Proof. In order to deny a signature, given $(\alpha, \alpha_1, \alpha_2, \beta_1)$ dishonest signer must find four secret keys $\bar{k}_1, \bar{k}_2, \bar{k}_3, \bar{k}_4$ such as

$$\begin{aligned} \alpha_1 &= (\bar{k}_3\alpha_1 + \bar{k}_1\beta_1)(\bmod q) \text{ and} \\ \alpha_2 &= (\bar{k}_4\alpha + \bar{k}_2\beta_1)(\bmod q) \end{aligned}$$

However, finding those secrete keys is hard to solve because it is difficult to solve elliptic curve discrete logarithm problems.

Lemma 4.8 *Different secret keys that match with the public key and pass the verification test for a message* m *create different* $\bar{m} \neq m$.

Proof. As shown in Theorem (3), an enemy with unlimited computational power can obtain q secret keys. Say, there is another signature which also passes the verification test, then we have

$$\begin{aligned} \bar{y}_1 &= (k_1\bar{m} + k_2)(mod\, q) \\ \bar{y}_2 &= (k_3\bar{m} + k_4)(mod\, q) \end{aligned}$$

and obtain the following equation

$$\begin{pmatrix} c & 0 & 1 & 0 \\ 0 & c & 0 & 1 \\ m & 1 & 0 & 0 \\ 0 & 0 & m & 1 \\ \bar{m} & 1 & 0 & 0 \\ 0 & 0 & \bar{m} & 1 \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ y_1 \\ y_2 \\ \bar{y}_1 \\ \bar{y}_2 \end{pmatrix} (\bmod q)$$

The coefficient matrix has rank 4 and the equations contain 4 variables. Thus there is only one unique solution. Theorems 4.4, 4.5 and Lemma 4.7 show that the proposed scheme satisfies all requirements of fail-stop signature mentioned in Section 2.

5. Efficiency performance

The performance of our scheme is described in terms of number of keys, computational complexity. We use the following notations to analyze the performance of our scheme.

Table 1: The comparison among our scheme and Esmail and Yahya scheme

Items	scheme by Eddie and Yahya		Our scheme	
	Time complexity	Complexity in T_{mul}	Time Complexity	Complexity in T_{mul}
Key generation	$8T_{exp} + 3T_{mul}$	$1923T_{mul}$	$8T_{ec-mul}+$ $3T_{ec-add}$	$232.36T_{mul}$
Signature generation	$4T_{mul}$	$4T_{mul}$	$4T_{ec-mul}+$ $2T_{ec-add}$	$32.24T_{mul}$
Signature verification	$4T_{exp} + 2T_{mul}$	$962T_{mul}$	$4T_{ec-mul}+$ $2T_{ec-add}$	$232.36T_{mul}$

- T_{mul} is the time complexity for executing the modular multiplication,
- T_{exp} is the time complexity for executing the modular exponentiation,
- T_{ec-add} is the time complexity for executing the addition of two points in elliptic curve,
- T_{ec-mul} is the time complexity for executing the multiplication of a number and an elliptic curve,
- T_h is the time complexity for performing a one-way hash function

We ignore the negligible time performing for modular addition. The computational complexity for the key generation, signing generation and verification is given in Table 1 and the last column converts various operation units to T_{mul} where $T_{exp} \approx 40T_{mul}$, $T_{ec-mul} \approx 29T_{mul}$ and $T_{ec-add} \approx 0.12T_{mul}$ given by Koblitz, Menezes and Vanstone [9].

Ismail and Yahya proposed fail-stop designated signature scheme. In the scheme, the authors proved that fail-stop designated signature scheme proposed by them is superior in performance to other RSA-like schemes, but our scheme can be instantiated using elliptic curve arithmetic over small prime fields. This is important for many applications. We can seen in Table 1 that our scheme is mush more efficient than the Ismail and Yahya scheme [1] for key generation, signature and verification.

6. Numerical simulation of the scheme

Let say, a recipient Bob wishes to obtain a signature on his message $m = 603$. The scheme's setup is done by a trusted dealer, TD who generates the following parameters:

$$y^2 = (x^3 - 3x + 33)(\mathrm{mod}\, 1091)$$
$$p = 1091,\ q = 1051,$$
$$G = [299, 62] \text{ is a base with order } q$$
$$d = 233,\ \alpha = 195G = [401, 993]$$
$$\beta = d\alpha = 233(195G) = 242G = [4, 1080]$$

and TD sends β to signer Alice via secure channel but broadcasts the pair (α, q). To communicate, Bob and Alice must agree on a common secret key, say they choose $\lambda = 721$ at random. Alice then passes securely β to Bob and next Bob selects his random private key $v = 1021$ and computes and securely sends Alice

$$\gamma = v\beta = 1021(242G) = 97G = [795, 484]$$

Alice then proceeds by selecting his four secret keys in $GF(q)$. Let the keys be the following:

$$k_1 = 171, k_2 = 341, k_3 = 521 \text{ and } k_4 = 622$$

These keys must be used only once. This means to sign different message, Alice should selects another keys because if the same keys are used twice, the keys will be recovered easily. Now Alice will generate three public keys related to Bob. The keys are given by

$$\begin{aligned}\beta_1 &= 622(195G) + 521(97G) = 514G = [27, 759]\\ \alpha_1 &= 521(195G) + 171(514G) = 309G = [60, 933]\\ \beta_2 &= 622(195G) + 341(514G) = 182G = [759, 198]\end{aligned}$$

and will be used by Bob to validate the produced signature. Now, to sign $m = 603$, Alice computes the following:

$$\begin{aligned}y_1 &= k_1 m + k_2\lambda = 171(603) + 341(721) = 42\\ y_2 &= k_3 m + k_4\lambda = 521(603) + 622(721) = 650\end{aligned}$$

and these values are the signature of $m = 603$. To validate, only Bob has a right to do so. He checks that $y_2\alpha + y_1\beta_1 = m\alpha_1 + \lambda\alpha_2\,(over E_p(a, b))$ holds. Since

$$650(195G) + 42(514G) = 147G = 603(309G) + 721(182G)$$

Bob accepts the signature. It is very important for Bob to prove to any third party say, Johnson that the signature is valid otherwise Bob and the signature cannot be trusted. This is done via zero-knowledge techniques.

Now, say an enemy, Song claims to Johnson that $(\bar{y}_1, \bar{y}_2) = (1023, 53)$ is also a valid signature of 603 produced by Alice. Unfortunately, this is true since

$$53(195G) + 1023(514G) = 147G = 603(309G) + 721(182G)$$

and Johnson may brings this to court because he feels that Alice and Bob is trying to cheat him. To prove that they are not guilty, Alice and Bob must show that the private keys have been broken. To do this, Alice first generates a signature on the message 603 and this is given by $(42, 650)$. He then calculates the two numbers as below:

$$\begin{aligned} y_2 - \bar{y}_2 &= 650 - 53 = 597 \\ \bar{y}_1 - y_1 &= 1023 - 42 = 981 \end{aligned}$$

and next Alice cooperates with Bob to prove that a forgery has happened by compute the following :

$$\begin{aligned} d &= \frac{(y_2 - \bar{y}_2) - (\bar{y}_1 - y_1)k_4}{(\bar{y}_1 - y_1)k_3 v} (\text{mod} q) \\ &= \frac{597 - 981(622)}{981(521)1021} \equiv 1046(584) \equiv 233 (\text{mod}\, 1051) \end{aligned}$$

and use this as the proof of forgery.

7. Conclusion

In this article, we presented a new fail-stop designated recipient signature scheme based on elliptic curve discrete logarithm problems. with two properties: (1) only an intended recipient can validate the resulting signature and prove it to any requested third party and (2) if there was a forgery then both the signer and recipient can co-operate to provide a proof that a forgery has happened. The security of Ismail and Yahya fail-stop designated recipient signature scheme is constructed on the integer factorization problem while the security of the proposed scheme is based on the difficulty of solving the elliptic curve discrete logarithm problem. According to [15], the elliptic curve discrete logarithm problem is significantly more difficult than the integer factorization problem. For the most part, the well-known RSA system [21] must use 1024 bit keys, only then can it attain computationally reasonable security; the ECC needs only 160 bit keys. So, at the same level of security, the speed of ECC is several times faster than RSA system; it can also saves on key storage space.Therfore, we showed that our scheme is mush more efficient than the Ismail and Yahya scheme. Since implemented on the elliptic curve cryptosystem, the proposed scheme enables to reach the best trade-off between security and efficiency.

References

[1] Ismail, S.E., Yahya, A.H., *Fail-stop designated recipient signature scheme and its applications*, Matematika UTM23, vol. 23 (1) (2008), 9-21.

[2] Gerrit, B., Pfitzmann, B., Waidner, M., *A Remark on A Signature scheme where forgery can be proved,* In Advances in Cryptology Eurocrypt'90, LNCS 437, Springer-Verlag, 1991, 441-445.

[3] Goldwasser, S., Micali, S., Rivest, R.A., *Digital signature scheme secure against adaptive Chosen-Message Attack*, SIAM Journal on Computing, vol. 17 (2) (1988), 281-308.

[4] Heyst, E.V., Pedersen T., *How to Make efficient Fail-Stop Signatures*, In Advances in Cryptology-Eurocrypt'92, 1992, 337-346.

[5] Heijst, E.V., Pedersen, T., Pftizman, B., *New construction of fail stop signature and lower bounds,* In Advances in Cryptology Crypto'92, LNCS 740, 1993, 15-30.

[6] Koblitz, N., *Elliptic curve cryptosystems,* Mathematics of Computation, vol. 48 (177) (1987), 203209.

[7] Koblitz, N., *A Course in Number Theory and Cryptography,* Springer-Verlag, Berlin, 1994.

[8] Koblitz, N., *Algebraic Aspects of Cryptography*, Springer-Verlag, Berlin, 1997.

[9] Koblitz, N., Menezes, A., Vanstone, S., *The state of elliptic curve cryptography*, Design, Code Cryptograph, vol. 19 (2-3) (2000), 173-193.

[10] Lannis, C., Apstolos, P., Paul, G., Yannis, C., *Elliptic curve based zero knowledge proofs and their applicability on resource constrained devices*, International Conference on Mobile Adhoc and Sensr System (MASS), IEEE, 2011.

[11] Menezes, A.J., *Elliptic Curve Public Key Cryptosystem,* Kluwer Academic Publishers, 1993.

[12] Merkle, R.C., *Protocols for public key cryptosystem*, In Proc. Symposium on Security and Privacy, 1980, 122-134.

[13] Michael, W., Birgit, P., *The Dining Cryptographers in the Disco: Unconditional Sender and Recipient Untraceability with Computationally Secure Serviceability*, Eurocrypt'89, Lecture Notes in Computer Science 434, 1990.

[14] Miller, V.S., *Use of elliptic curves in cryptography,* Advances in cryptology Proceedings of Crypto '85, LNCS, vol. 218, Springer, 1986, 417426.

[15] NIST, DRAFT, Special Publication 800-57 (2003), Recommendation onkey management, January 2003, http://csrc.nist.gov/CryptoToolkit/kms/guideline-1-Jan03.pdf

[16] Pedersen, T.P., Pfitzmann, B., *Fail-stop signatures*, SIAM Journal on Computing, vol. 26 (2) (1997), 291-330.

[17] Pftizman, B., *Digital signature schemes General framework and fail-stop signatures*, Lecture Notes in Computer Science 1100, Springer-Verlag, 1996.

[18] Pftizman, B., *Sorting out signature schemes*, CWI Quaterly, 8/2 (1995), 147-172.

[19] Pftizman, B., *Fail-Stop Signature: Principles and applications*, Proc. Compsec'91, 8th World Conference on Computer Security, Audit and Control, Elseviar, Oxford, 1991, 125-134.

[20] Rene, S., *Counting points on elliptic over finite fields*, Journal de Theeorie des Nombres, Bordeaux, vol. 7 (1995), 219-254.

[21] Rivest, R.L., Shamir, A., Adleman, L., *A method for obtaining digital signatures and public-key cryptosystems*, Communications of The ACM, vol. 21 (2) (1978), 120126.

[22] Stison, D.R., *Cryptography: Theory and Practic*, CRC Press, Boca Raton, New York, 1995.

[23] Susilo, W., Naini-Safavi, R., Pieprzy. J., *RSA-Based Fail Stop Signature Schemes*, International Workshop on Security (IWSEC'99), IEEE Computer Society Press, 1999, 161-166.

[24] Tetsuya, I., Jun, K. Masayuku, N., Kazuhiro, Y.E., *Efficient Implementation of Schoof's Algorithm*, Asiacrypt '98, Lecture Notes in Computer Science 1519, 1998, 66-79.

Accepted: 04.07.2013