MATEMATIKA, 2007, Volume 23, Number 1, 9–21

Fail-Stop Designated Recipient Signature Scheme and its Applications

[1]**Eddie Shahril Ismail** & [2]**Yahya Abu Hasan**

[1]School of Mathematical Sciences, Faculty of Science & Technology
National University of Malaysia, 43600 UKM Bangi, Selangor, Malaysia

[2]School of Mathematical Sciences, University Science of Malaysia
11800 USM Minden, Penang, Malaysia.

e-mail: [1]esbi@ukm.my, [2]ahyahya@cs.usm.my

Abstract How to design a fail-stop signature scheme is still a field in need of cultivation. If an attacker can successfully break an ordinary fail-stop signature scheme, then only a signer not a recipient is able to prove that a forgery has happened. This system only solves one dimensional of cryptographic problems. In this paper, we describe a new kind of digital signature scheme called fail-stop designated recipient signature scheme. The scheme allows a signer and an intended recipient to cooperatively 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 prove the validity to any interested 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.

Keywords Cryptography, Digital Signature, RSA Signature Scheme, Discrete Logarithm Problem, Factorization Problem, Fail-Stop Signature and Zero-Knowledge Protocol

1 Introduction

Digital signature scheme allows an electronic document to be signed by a signer in a signing algorithm. Later, an owner of the document can determine whether the resulting signature is valid or not to confirm its integrity by performing a verifying algorithm. Up to now, many types of signature schemes have been invented by Camenisch[4], Chaum[5] and Lim[10] and each of them has a unique feature so that it can solve a unique problem in the real world situations. Basically, in all classical digital signature schemes if a forger successfully obtain a forge signature that passes the verification procedure then the scheme is completely insecure and a signer would be blamed then. This is the main disadvantage of the digital signature scheme. However, fail-stop signature completely avoids this type of attack. If an attacker successfully forges a signature, then a real signer can provide a proof and show to any third party that a forgery has happened. This concept was first introduced by Waidner

and Pfitzmann[20] and then was formally defined in Pfitzmann and Waidner[17]. Theoretically, fail-stop signature schemes are known to exist if claw-free pairs of permutations (not necessarily with trapdoor) exist, see Bleumer et. al[2] and Pfitzmann and Waidner[18] for description and Pfitzmann and Waidner[17] for proof. For complete definition of claw-free permutations, see Goldwasser et. al[7]. In particular, Bleumer et. al [2] and Heijst et. al[8] show that fail-stop signatures only exist if factoring large integer or discrete logarithm problem is hard.

The first construction of fail-stop signature by Waidner & Pfitzmann[20] uses a one-time signature scheme and requires messages to be signed bit by bit although the tree-authentication as proposed by Merkle[11] 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 (see Pfitzmann[14]). 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. Pfitzmann[15] also presented an efficient scheme with single recipient. First fail-stop signature schemes based on factorization and discrete logarithm problems have been proposed respectively in Susilo et. al[19] and Heyst and Pedersen[9]. The former uses modulus factors as a proof of forgery whereas the latter uses a private key of the system as a proof of forgery. In Pedersen and Pfitzmann[13], a formal definition of fail-stop signature is given and a general construction using bundling homomorphism is proposed. See also Pfitzmann[16]. To provide a proof of forgery, they show that under bundling homomorphism, two signatures (respectively generated by signer and forger) are shown to collide. Note that, none of the proposed schemes provide that, only an intended recipient can verify the signature and the proof of forgery must be done collaboratively by the two entities: the signer and the designated recipient. These properties if satisfied are very useful in many various applications that we will be discussed later.

In this paper, we propose a new digital signature called Fail-Stop Designated Recipient Signature (FDRS) scheme. The scheme allows a signer and an intended recipient to cooperatively 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 it is possible to solve certain problems concerning the protection of confidential documents especially those that are personally sensitive to the owner of documents. As usual the scheme will involve three players: a signer who issues a signature, a designated recipient as a recipient of signatures and an adversary (Adv) who always attempting to forge signatures.

1.1 Problem Statement

If an attacker can successfully break an ordinary fail-stop signature scheme, then only a signer not a recipient is able to prove that a forgery has happened plus outsiders can verify the resulting signature even though the signed document is very confidential to the actual owner. This should be avoided and to overcome this problem we create FDRS such that only an intended verifier is able to verify his or her confidential document. Also, the intended verifier can later prove to any third party that the resulted signature is genuine.

The paper is organized as follows. In the next section, we present the basic concepts and definitions of our model of fail-stop designated recipient signature schemes. In section

3, we present our new scheme and prove that it satisfies the requirements of the model in section 4. We next discuss some immediate applications of our scheme and present a small example and this can be found in section 5. Finally, section 6 concludes the paper.

2 The Model of Fail-Stop Designated Recipient Signature (FDSR)

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

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

(i) GEN, SIGN and VER are respectively for generating keys, signing messages and verifying signatures.

(ii) PVER for recipient to prove validation of signature to a third party without revealing any secret information.

(iii) 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:

A1. If the signer signs a message, then the recipient can verify the signature and accept it as genuine.

A2. A polynomially bounded (limited) forger cannot create forged signatures that successfully pass the VER.

A3. 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.

A4. A polynomially bounded (limited) signer or recipient cannot create a signature that he can later prove to be a forgery.

To achieve the above properties, we must have:

B1. For each public key, there must exist many matching secret keys. In other words, there must exist SK, a set consisting of secret keys and each of which fits with the signer's public key, PK.

B2. Different secret keys from SK must create different signatures on the same message, m.

B3. The real signer knows only one of the secret keys and thus can only construct one possible signature on a given message.

B4. 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:

C1. 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.

C2. 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.

C3. A polynomially bounded signer cannot construct a 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 others 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 former.

3 The New Scheme

We now present our fail-stop designated recipient signature scheme based on the factorization problem and discrete logarithm problem using a scheme of Susilo et. al [19] as our basic scheme.

The System Setup (GEN):
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:

(i) Choose two 512-bit primes p and q such that $p = 2\bar{p} + 1$ and $q = 2\bar{q} + 1$ where $\bar{p}$ and $\bar{q}$ are also prime numbers.

(ii) Compute $n = pq$ and $\varphi(n) = (p-1)(q-1)$.

(iii) Choose an integer $\alpha \in \mathrm{Z}_n^* = \{1 \leq \upsilon \leq n-1 | \gcd(\upsilon, n) = 1\}$ arbitrarily and a secret key $d \in \mathrm{Z}_{\varphi(n)}^* = \{1 \leq \eta \leq \varphi(n)-1 | \gcd(\eta, \varphi(n)) = 1\}$.

(iv) Compute $e \equiv d^{-1} \bmod \varphi(n)$ and $\beta \equiv \alpha^d \bmod n$.

(v) Broadcast (α, n) and send the pair (e, β) to the signer via secure channel.

A signer then identifies an intended recipient who wishes to accept his or her resulting signature. They then agree on a random common secret key $\lambda \in_R \mathrm{Z}_n^*$. Next the signer passes securely the integer β to the intended recipient who then does the following:

(vi) Select at random private key $x_R \in_R \mathrm{Z}_n^*$ and compute and send $\gamma \equiv \beta^{x_R} (\mathrm{mod}\, n)$ securely to the signer.

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

(vii) Choose four secret keys $k_1, k_2, k_3, k_4 \in \mathrm{Z}_n^*$.

(viii) Compute the following public keys that are related to the intended recipient:

$$\beta_1 \equiv \alpha^{k_4}\gamma^{k_3} (\mathrm{mod}\, n),$$

$$\alpha_1 \equiv \alpha^{k_3}\beta_1^{k_1} (\mathrm{mod}\, n) \text{ and } \alpha_2 \equiv \alpha^{k_4}\beta_1^{k_2} (\mathrm{mod}\, n). \quad (1)$$

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.

Signing (SIGN) and Verifying (VER):

To sign a message $m \in \{0,1\}^*$, the signer is implemented according to the following steps:

(i) Compute $y_1 = k_1 m + k_2 \lambda$ and $y_2 = k_3 m + k_4 \lambda$.

(ii) A valid signature on a message m is given by (y_1, y_2).

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

$$\alpha^{y_2}\beta_1^{y_1} \equiv \alpha_1^m \alpha_2^\lambda (\mathrm{mod}\, n). \quad (2)$$

Note that only the intended recipient, who knows λ, can verify the signature. Unfortunately, the signer also has an ability to check the validity of signature, but this is not a serious problem when the scheme is applied to our applications.

Prove of Validation (PVER):

In some cases, the recipient needs to prove the validity of the signature to any third party. To do this, the signer computes and sends recipient $\omega_1 = \alpha^{k_1}$ and $\omega_2 = \alpha^{k_2}$ secretly. The recipient then sends third party $\omega_3 = \alpha_2^\lambda$ whose next confirms that

$$\alpha^{y_2}\beta_1^{y_1}\alpha_1^{-m} \equiv \omega_3 (\mathrm{mod}\, n). \quad (3)$$

Finally the recipient proves that $\log_{\omega_2} \alpha^{y_1}\omega_1^{-m} = \log_{\alpha_2} \alpha^{y_2}\beta_1^{y_1}\alpha_1^{-m}$ in a zero-knowledge technique for example using the protocol for simultaneous discrete logarithm [3, 6].

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.

Algorithm Proof of Forgery (PTEST):

(i) Signer constructs his signature on m as (y_1, y_2).

(ii) Signer then calculates $Z_1 = \bar{y}_1 - y_1$ and $Z_2 = y_2 - \bar{y}_2$.

(iii) Signer and recipient jointly compute $Z = e(Z_2 - k_4 Z_1) - x_R k_3 Z_1 = c\varphi(n)$ for some integer c.

With the knowledge of $c\varphi(n)$ and the modulus n, the signer or the intended recipient can find the non-trivial factors of n using Miller-Bach algorithm [12, 1]. Note that, both signer and recipient are required to provide a proof of forgery. It is also attractive if everyone can provide a proof of forgery with signer's and recipient's consent if both of them are not available.

Theorem 3.1 *If* GEN *and* SIGN *are run smoothly then the validation of signature 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:

$$\alpha^{y_2}\beta_1^{y_1} \equiv \alpha^{k_3 m + k_4 \lambda}\beta_1^{k_1 m + k_2 \lambda} \equiv \left(\alpha^{k_3}\beta_1^{k_1}\right)^m \left(\alpha^{k_4}\beta_1^{k_2}\right)^{\lambda} \equiv \alpha_1^m \alpha_2^{\lambda} (\mathrm{mod}\, n). \tag{4}$$

To illustrate the above scheme, we will describe it using a simple flowchart as illustrated in Figure 4.1. The flowchart contains all algorithms involved in this scheme.

4 Security and Efficiency Performances

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, to provide 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* $\varphi(n)^2$ *equally likely secret keys that match with the signer-related public key.*

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

$$\begin{aligned} \alpha_1 &\equiv \alpha^{k_3}\beta_1^{k_1} (\mathrm{mod}\, n), \\ \alpha_2 &\equiv \alpha^{k_4}\beta_1^{k_2} (\mathrm{mod}\, n), \\ \beta_1 &\equiv \alpha^{k_4}\gamma^{k_3} (\mathrm{mod}\, n) \end{aligned} \tag{5}$$

and by denoting $z = \log_\alpha \beta_1 = k_4 + dx_R k_3$, the above equations (the first two) can be written as

$$\alpha_1 \equiv \alpha^{k_3 + k_1 z} \bmod n \text{ and } \alpha_2 \equiv \alpha^{k_4 + k_2 z} \bmod n \tag{6}$$

By solving the discrete logarithm problem, we obtain

$$z_1 \equiv (k_3 + k_1 z) \bmod \varphi(n) \text{ and } z_2 \equiv (k_4 + k_2 z) \bmod \varphi(n) \tag{7}$$

for $z_1, z_2 \in \mathrm{Z}_n^*$ or equivalently we have:

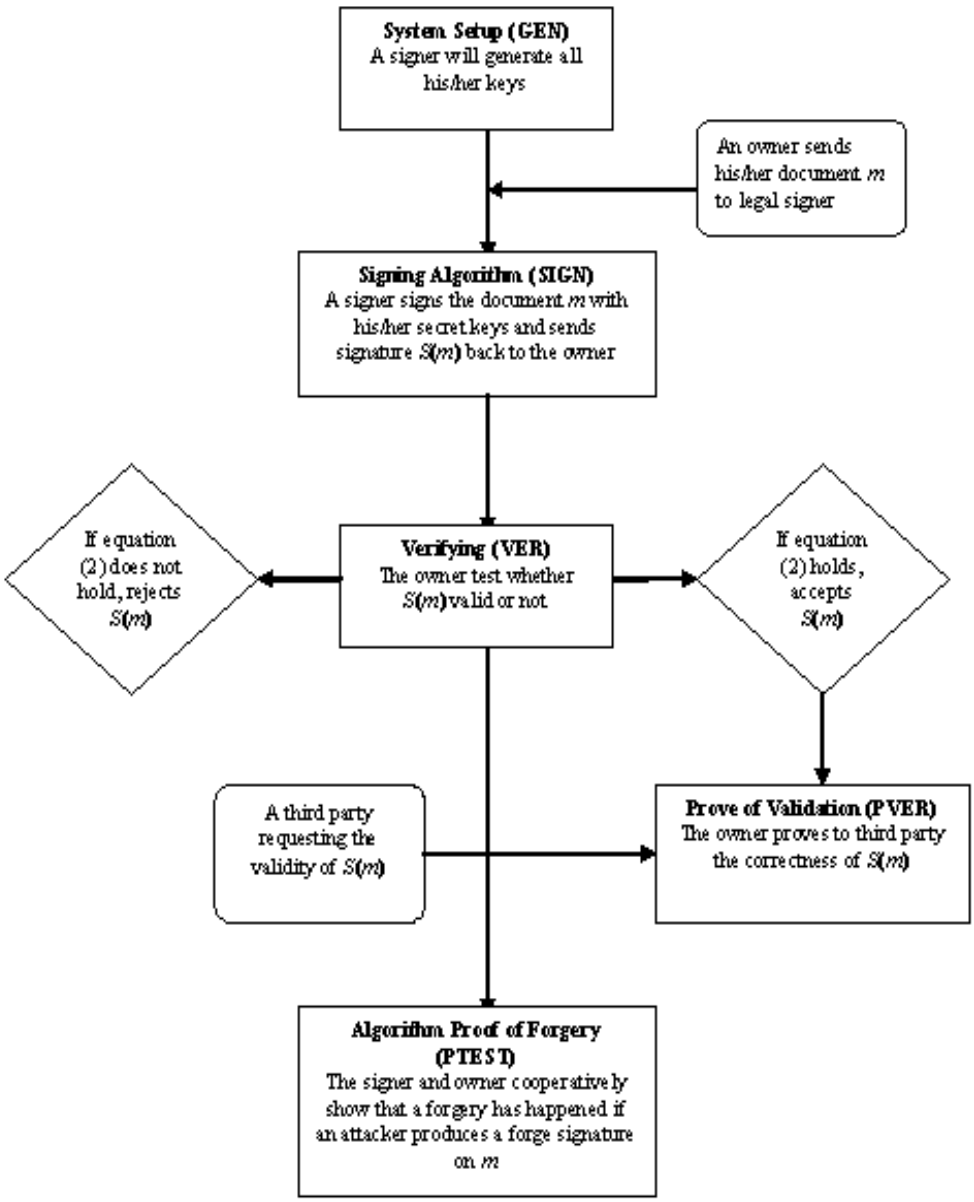

Figure 1: Flowchart of the FRDS

$$\begin{pmatrix} z & 0 & 1 & 0 \\ 0 & z & 0 & 1 \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \end{pmatrix} = \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \bmod \varphi(n) \tag{8}$$

These equations contain 4 variables and the rank of the coefficient matrix is 2. We see that, k_3 and k_4 can take values between 1 to $\varphi(n)$ thus we can obtain $\varphi(n)^2$ different solutions.

Lemma 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 or she can factorize the modulus* n.

Proof: Since both signatures (y_1, y_2) and $(\bar{y}_1, \bar{y}_2)$ pass the verification test then according (1) we must have that

$$\begin{aligned} \alpha^{y_2}\beta_1^{y_1} &\equiv \alpha^{\bar{y}_2}\beta_1^{\bar{y}_1} (\bmod\, n) \\ \alpha^{y_2}\left(\alpha^{k_4}\gamma^{k_3}\right)^{y_1} &\equiv \alpha^{\bar{y}_2}\left(\alpha^{k_4}\gamma^{k_3}\right)^{\bar{y}_1} (\bmod\, n) \\ \alpha^{y_2+y_1k_4+dx_Rk_3y_1} &\equiv \alpha^{\bar{y}_2+\bar{y}_1k_4+dx_Rk_3\bar{y}_1} (\bmod\, n) \\ y_2 + y_1k_4 + dx_Rk_3y_1 &\equiv \bar{y}_2 + \bar{y}_1k_4 + dx_Rk_3\bar{y}_1 (\bmod \varphi(n)) \\ (y_2 - \bar{y}_2) + (y_1 - \bar{y}_1)k_4 &\equiv dx_Rk_3(\bar{y}_1 - y_1) (\bmod \varphi(n)). \end{aligned} \tag{9}$$

Note that at this stage the signer or recipient does not know $\varphi(n)$. However, by multiplying both sides with e and using $Z_1 = \bar{y}_1 - y_1$ and $Z_2 = y_2 - \bar{y}_2$, we then have the following:

$$e(Z_2 - Z_1k_4) \equiv x_Rk_3Z_1 (\bmod \varphi(n)) \tag{10}$$

or equivalently $Z = e(Z_2 - k_4Z_1) - x_Rk_3Z_1 = c\varphi(n)$ for some integer c. We have completed the proof.

Theorem 4.1 *Knowing the public key together with the signature for a message* m*, an enemy unlimited computational power can calculate* $\varphi(n)$ *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 discrete logarithm and factorization problems. The enemy particularly has the following equations:

$$\begin{aligned} z_1 &\equiv (k_3 + k_1z) \bmod \varphi(n), \\ z_2 &\equiv (k_4 + k_2z) \bmod \varphi(n), \\ \bar{y}_1 &\equiv (k_1m + k_2) \bmod \varphi(n), \\ \bar{y}_2 &\equiv (k_3m + k_4) \bmod \varphi(n) \end{aligned} \tag{11}$$

where $z = \log_\alpha \beta_1 = k_4 + dx_Rk_3$, $z_1, z_2 \in \mathrm{Z}_n^*$ and the last two equations is an acceptable signature on m. These equations next can be formed as

$$\begin{pmatrix} z & 0 & 1 & 0 \\ 0 & z & 0 & 1 \\ m & 1 & 0 & 0 \\ 0 & 0 & m & 1 \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \end{pmatrix} = \begin{pmatrix} z_1 \\ z_2 \\ \bar{y}_1 \\ \bar{y}_2 \end{pmatrix} \bmod \varphi(n). \tag{12}$$

The coefficient matrix has rank 3 and the equations contain 4 variables. Thus there are exactly $\varphi(n)$ solutions to this equation.

Corollary 4.1 *An enemy with unlimited computational power cannot compute the signer's signature on a new message.*

Proof: From Theorem 4.1, the enemy knows $\varphi(n)$ secret keys but unable to determine which key will be used by signer.

The following theorem guarantees no limited computational power of signers to be able to deny his or her signatures.

Theorem 4.2 *A computationally bounded signer cannot make signatures, which he or she can later prove to be forgeries.*

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

$$\alpha_1 \equiv \alpha^{\bar{k}_3} \beta_1^{\bar{k}_1} (\bmod\, n) \text{ and } \alpha_2 \equiv \alpha^{\bar{k}_4} \beta_1^{\bar{k}_2} (\bmod\, n) \tag{13}$$

However, finding those secret keys is equivalent to the discrete logarithm problem, which is hard to solve.

Lemma 4.3 *Different secret keys that match with the public key and pass the verification test for a message m create different signatures on $\bar{m} \neq m$.*

Proof: As shown in Theorem 4.1, an enemy with unlimited computational power can obtain $\varphi(n)$ secret keys. Say, there is another signature $(\bar{y}_1, \bar{y}_2)$ which also passes the verification test then we have

$$\begin{aligned} \bar{y}_1 &\equiv (k_1 \bar{m} + k_2) \bmod \varphi(n) \\ \bar{y}_2 &\equiv (k_3 \bar{m} + k_4) \bmod \varphi(n) \end{aligned} \tag{14}$$

and obtain the following equation

$$\begin{pmatrix} z & 0 & 1 & 0 \\ 0 & z & 0 & 1 \\ m & 1 & 0 & 0 \\ 0 & 0 & m & 1 \\ \bar{m} & 1 & 0 & 0 \\ 0 & 0 & \bar{m} & 1 \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \end{pmatrix} = \begin{pmatrix} z_1 \\ z_2 \\ y_1 \\ y_2 \\ \bar{y}_1 \\ \bar{y}_2 \end{pmatrix} \bmod \varphi(n). \tag{15}$$

The coefficient matrix has rank 4 and the equations contain 4 variables. Thus there is only one unique solution.
For efficiency aspect, our scheme has no difference with the scheme presented in [19]. In that scheme, to sign a message, the signer has to use different secret key for different message. This means for each sender of a message, the signer's public and secret key will

be periodically generated. Our scheme also suffers from this problem. Our SIGN has no modular exponentiation but involves two standard multiplications and VER involves only three modular exponentiations of modulo n.

5 Applications Through Case Study

We discuss two immediate applications corresponding to our FDRS. They are: protection of confidential document and flexible shared verification. For the former, we discuss it through a case study.

(I) Protection of Confidential Documents

Using the new signature scheme, we can protect our confidential documents (or document that personally sensitive). For example, a document contains some required information that qualifies the owner to vote electronically, a health certification required by University before a student registers as a graduate student or before a person can be employed as an accountant and a certificate contains information of qualifications before allowing a candidate to be interviewed. These are amongst the confidential documents that contain some parts of the owner information. The protection of these documents are obviously important, to avoid any accusation of integrity of a document, owner and the party who issues a signature on the document.

Our case study consists of the following: Consider the following situation for three entities: A recipient Remy, Medical Varsity - an entity that produces signatures and World Medic - an interested third party. Remy graduated in Optometry, wishes to find a job at World Medic as an Optometrist. He then attends a virtual interview and was asked to 'present' his certificate (diploma, degree, etc) that has issued and signed by Medical Varsity. He then was offered the post. Sometimes later, World Medic accepts a forged signature on Alice's certificate. World Medic starts to investigate and brings the case to court. However, Remy and Medical Varsity prove that they are not guilty by showing that a forgery has happened by performing the FDRS.

Let say, a recipient Remy wishes to obtain a signature on his certificate $m = 808$, from an officer of Medical Varsity, Shawn in a FDRS. The scheme's setup is done by a trusted dealer, TD who generates the following parameters:

$$\begin{gathered} p = 1319, q = 383, \\ n = pq = 505177, \varphi(n) = (p-1)(q-1) = 503476, \\ \alpha = 11922, d = 3799, \\ e \equiv d^{-1} \equiv 3799^{-1} \equiv 338743 \bmod 503476, \\ \beta \equiv \alpha^d \equiv 11922^{3799} \equiv 498165 \bmod 505177 \end{gathered}$$

and TD sends Shawn $(e, \beta) = (338743, 498165)$ via secure channel but broadcasts the pair $(\alpha, n) = (11922, 505177)$. To communicate, Remy and Shawn must agree on a common secret key, say they choose $\lambda = 764$ at random. Shawn then passes securely β to Remy and next Remy selects his random private key $x_R = 7998$ and computes and securely sends Shawn

$$\gamma \equiv \beta^{x_R} \equiv 498165^{7998} \equiv 219816 \bmod 505177.$$

Shawn then proceeds by selecting his four secret keys in Z_n^*. Let the keys be the following:

$$k_1 = 321, k_2 = 456, k_3 = 234 \text{ and } k_4 = 127.$$

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

$$\begin{aligned}\beta_1 &\equiv (11922)^{127}(219816)^{234} \equiv 67198 \bmod 505177\\ \alpha_1 &\equiv (11922)^{234}(67198)^{321} \equiv 220058 \bmod 505177\\ \alpha_2 &\equiv (11922)^{127}(67198)^{456} \equiv 303464 \bmod 505177\end{aligned}$$

and will be used by Remy to validate the produced signature. Now to sign 808, Shawn computes the following:

$$\begin{aligned}y_1 &= 321(808) + (456)(764) = 607752\\ y_2 &= 234(808) + (127)(764) = 286100\end{aligned}$$

and these values are the signature of 808. To validate, only Remy has a right to do so. He checks that $\alpha^{y_2}\beta_1^{y_1} \equiv \alpha_1^m\alpha_2^\lambda \bmod n$ holds. Since

$$(11922)^{286100}(67198)^{607752} \equiv 172048 \equiv (220058)^{808}(303464)^{764} \bmod 505177,$$

Remy accepts the signature. It is very important for Remy to prove to any third party say, Chen that the signature is valid otherwise Remy and the signature cannot be trusted. This is done via zero-knowledge techniques.
Now say an enemy, Eve claims to Chen that $(44347, 3)$ is also a valid signature of 808 produced by Shawn. Unfortunately, this is true since

$$(11922)^3(67198)^{44347} \equiv 172048 \equiv (220058)^{808}(303464)^{764} \bmod 505177$$

and Chen may brings this to court because he feels that Shawn and Remy is trying to cheat him. To prove that they are not guilty, Shawn and Remy must show that the primes p and q have been broken. To do this, Shawn first generates a signature on the message 808 and this is given by (607752, 286100). He then calculates the two numbers as below:

$$\begin{aligned}z_1 &= 44347 - 607752 = -563405\\ z_2 &= 286100 - 3 = 286097\end{aligned}$$

and next Shawn cooperates with Remy to calculate

$$\begin{aligned}z &= e(z_2 - k_4 z_1) - x_R k_3 z_1\\ &= 338743(286097 - (127)(-563405)) - (7998)(234)(-563405)\\ &= 25389230331736\end{aligned}$$

and this number can be used to find p and q (using Miller-Bach algorithm) because

$$z = 25389230331736 = (50427886)(503476) = c\varphi(n)$$

where $c = 50427886$.

(II) Flexible Shared Verification

The fail-stop designated recipient also can be immediately extended to the flexible shared verification such that two or more recipients can jointly verify any resulting signature and cooperatively provide a proof of forgery with a signer. Consider two recipients A and B with their respective secret key u and v wish to verify a signature jointly. To achieve this, they just compute and send $\gamma \equiv \beta^{u+v} (\mathrm{mod}\, n)$ securely to a signer. The rest of procedures are similar.

6 Conclusion

We have presented a model and example of our fail-stop designated recipient signature scheme 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. These properties result two immediate applications, protection of confidential documents and flexible shared verification. We believe our new kind of signature scheme will provide a better service than the original fail-stop signature scheme.

References

[1] E. Bach, *Discrete Logarithm and Factoring*. Report no. UCB/CSD 84/186, Comp. Sc. Division (EECS), University of California, Berkeley,(1984).

[2] G. Bleumer, B. Pfitzmann, & M. Waidner, *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] J. Boyar, D. Chaum, I. Damgard ,& T. Pedersen, *Convertible Undeniable Signatures*. In Advances in Cryptology-Crypto'90, LNCS 537, Springer-Verlag, (1991), 189-205. D. Chaum, *Zero-Knowledge Undeniable Signatures*. In Advances in Cryptology-Eurocrypt'90, LNCS 473, Springer-Verlag, (1991), 458-464.

[4] J. Camenisch, *Efficient and Generalized Group Signatures*. In Advances in Cryptology-Eurocrypt '97, LNCS 1233, Springer-Verlag, (1997), 465-479.

[5] D. Chaum, *Blind Signature Systems*. In Proc. CRYPTO 83, Chaum, D (ed), New York, Plenum Press, (1984),153-153.

[6] D. Chaum, J. H. Evertse ,& J. van de Graff. *An Improved Protocol for Demonstrating Possession of Discrete Logarithms and Some Generalizations*. In Advances in Cryptology-Eurocrypt'87 Proceedings, Springer-Verlag, (1988a),127-141.

[7] S. Goldwasser, S. Micali ,& R. Rivest, *A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attack*. SIAM Journal on Computing, 17(2)(1988), 281-308.

[8] E. V. Heijst, T. Pedersen ,& B. Pfitzmann, *New Construction of Fail-Stop Signatures and Lower Bounds*. In Advances in Cryptology-Crypto'92, LNCS 740, (1993),15-30.

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

[10] C. H. Lim , P. J. Lee, *Directed Signatures and Applications to Threshold Cryptosystem.* Workshop on Security Protocol, Cambrige, (1996), 131-138.

[11] R. C. Merkle, *Protocols for Public Key Cryptosystems.* In Proc. 1980 Symposium on Security and Privacy,(1980), 122-134.

[12] G. L. Miller,*Riemann's Hypothesis and Tests for Primality.* Journal of Computer and System Sciences, 13 (1976), 300-317.

[13] T. Pedersen, & B. Pfitzmann, *Fail-Stop Signatures.* SIAM Journal on Computing, 26/2 (1997), 291-330.

[14] B. Pfitzmann,*Fail-Stop Signatures: Principles and Applications.* Proc. Compsec'91, 8th World Conference on Computer Security, Audit and Control, Elseviar, Oxford,(1991), 125-134.

[15] B. Pfitzmann, *Sorting Out Signature Schemes.* CWI Quaterly, 8/2 (1995), 147-172.

[16] B. Pfitzmann, *Digital Signature Schemes-General Framework and Fail-stop Signatures.* LNCS 1100, Springer-Verlag, (1996) .

[17] B. Pfitzmann, & M. Waidner, *Formal Aspects of Fail-Stop Signatures.* Interner Bericht, Fakultat fur Informatik, University Karlsruhe, Report 22/90, (1990).

[18] B. Pfitzmann, & M. Waidner, *Fail-Stop Signatures and Their Application.* Securicom 91, Paris, (1991), 145-160.

[19] W. Susilo, R. Safavi-Naini , & J. Pieprzyk, *RSA-Based Fail-Stop Signature Schemes.* International Workshop on Security (IWSEC'99), IEEE Comp. Soc. Press, (1999), 161-166.

[20] M. Waidner, & B. Pfitzmann, *The Dining Cryptographers in the Disco: Unconditional Sender and Recipient Untraceability With Computationally Secure Serviceability.* In Advances in Cryptology-Eurocrypt'89, LNCS 434. Springer-Verlag, (1990), 690.