A SM2 Elliptic Curve Threshold Signature Scheme without a Trusted Center

Yan Jie[1], Lu Yu[1], Chen Li-yun[2] and Nie Wei[3]
[1] Equipment Command and Management Department, Mechanical Engineering College
Shijiazhuang, Hebei 050003 - China
[e-mail: jack20030552@163.com]
[2] Information Engineering Department, Mechanical Engineering College
Shijiazhuang, Hebei 050003 - China
[e-mail: 619603185@qq.com]
[3] College of Information Engineering, Shenzhen University
Shenzhen, Guangdong 518060 - China
[e-mail: 15233112877@139.com]
*Corresponding author: Nie Wei

Received August 6, 2015; revised November 13, 2015; accepted December 10, 2015; published February 29, 2016

Abstract

Threshold signature is very important in identity authentication and some other applications. In December 2010, Chinese Encryption Administration released the SM2 elliptic curve digital signature algorithm as the first standard of the digital signature algorithm in China. At present, the papers on the threshold signature scheme based on this algorithm are few. A SM2 elliptic curve threshold signature scheme without a trusted center is proposed according to the Joint-Shamir-RSS algorithm, the Joint-Shamir-ZSS algorithm, the sum or diff-SS algorithm, the Mul-SS algorithm, the Inv-SS algorithm and the PM-SS algorithm. The proposed scheme is analyzed from correctness, security and efficiency. The correctness analysis shows that the proposed scheme can realize the effective threshold signature. The security analysis shows that the proposed scheme can resist some kinds of common attacks. The efficiency analysis shows that if the same secret sharing algorithms are used to design the threshold signature schemes, the SM2 elliptic curve threshold signature scheme will be more efficient than the threshold signature scheme based on ECDSA.

Keywords: Threshold signature; SM2 elliptic curve digital signature; secret sharing; without a trusted center

This research was supported by the National Natural Science Foundation of China [No. 61271152] and the Natural Science Foundation of Hebei Province, China [No. F2012506008].

http://dx.doi.org/10.3837/tiis.2016.02.025 **ISSN : 1976-7277**

1. Introduction

With the development of E-Commerce and E-Government, digital signature becomes more and more important. In some applications, more than a specified number of members need to accomplish the effective signature together, and anyone receiving the signature can validate the effectiveness of the signature by the public key. Threshold signature can satisfy the needs of these applications. Threshold signature is the important research content in threshold cryptography and it was first proposed by Boyd [1] and Desmedt [2]. In 1991, Desmedt and Frankel [3] designed the first (t,n) threshold signature scheme based on RSA digital signature algorithm by threshold secret sharing algorithm. In the (t,n) threshold signature scheme, the trusted center distributes the private key shares to the signers. At least t signers accomplish the signature together by their private key shares, which works out the problem of authority abuse well. Later on, threshold signature attracted the attention of the researchers and many research production were achieved. Harn [4] proposed a threshold signature scheme without a trusted center based on ElGamal signature algorithm. Gennaro et al. [5] proposed the sharing algorithm of the multiplication of two secrets and the sharing algorithm of the inverse of the secret, and proposed a DSS threshold signature scheme on this basis. Wang et al. [6] proposed a threshold signature scheme with a trusted center based on the intractability of the discrete logarithm problem. Miyazaki [7] proposed a threshold signature scheme based on the elliptic curve ElGamal cryptosystem. HAN Jin-rong et al. [8] proposed a (t,n) verifiable threshold signature scheme based on the Nyberg Ruepple elliptic curve signature algorithm and the Pedersen verifiable secret sharing algorithm. Peng Hua-xi et al. [9] introduced the concept of forward security into the threshold signature scheme based on the bilinear pairing, and proposed a forward secure threshold signature scheme based on the bilinear pairing. SHEN Zhong-hua et al. [10] proposed a new (t,n) threshold signature scheme based on the multivariate linear polynomial according to the ElGamal public key cryptosystem and Schnorr signature algorithm. According to the standard lattice problem, Slim Bettaieb et al. [11] and Qingbin Wang et al. [12] proposed an improved lattice-based threshold signature scheme respectively. Fei Li et al. [13] proposed a new identity-based threshold signature scheme from bilinear pairings and the proposed scheme has the advantages in efficiency and functionality. Jung Yeon Hwang et al. [14] proposed an enhanced (t,n) threshold signature scheme based on the Computational Diffie-Hellman Problem. Raman Kumar et al. [15] proposed an enhanced secure threshold signature scheme based on RSA cryptosystem. In the proposed scheme, both the combiner and the secret share holder can verify the correctness of the information that they receive from each other.

The elliptic curve cryptosystem was proposed in 1986, which is better than traditional cryptosystem (such as RSA and DSA cryptosystem) in security and efficiency. It has become the mainstream algorithm of public key algorithms gradually. In December 2010, in order to satisfy the needs of the identity authentication in E-Commerce and E-Government, Chinese Encryption Administration [16] released the SM2 elliptic curve public key algorithms including SM2 elliptic curve digital signature algorithm. As the first standard of public key algorithm released by Chinese government, the SM2 elliptic curve public key algorithms play an important role in the construction of Chinese information security system. At present, the papers on the threshold signature scheme based on SM2 elliptic curve digital signature

algorithm are few, so a SM2 elliptic curve threshold signature scheme without a trusted center is proposed in this paper.

2. SM2 Elliptic Curve Digital Signature Algorithm

First, we assume that the parameters of the SM2 elliptic curve public key algorithms include $GF(p)$, E, G, p and q. $GF(p)$ is a finite field, E is an elliptic curve over $GF(p)$, $G=(x_G, y_G)$ is a base point on the elliptic curve E (where G has a large order q), p and q are two large prime numbers. d is the private key of the user, where $d \in [1, q-1]$. $P=dG$ is the public key of the user. $h(\cdot)$ is a Hash function.

The signature process of SM2 elliptic curve digital signature algorithm is described as follows.

Step 1. The signer chooses a random number $k \in [1, q-1]$, then computes $kG=(x_1, y_1)$.

Step 2. For the message m to be signed, the signer computes $r=(h(m)+x_1) \bmod q$. If $r=0$ or $r+k=q$, the signer will go back to Step 1.

Step 3. The signer computes $s=(1+d)^{-1}(k-rd) \bmod q$. If $s=0$, the signer will go back to Step 1, else he will get the signature (r,s) of the message m.

The signature validation process of SM2 elliptic curve digital signature algorithm is described as follows.

Step 1. When the verifier receives the message m and its signature (r,s), he will validate whether (r,s) satisfy $r,s \in [1, q-1]$ and $r+s \neq q$ or not. If (r,s) doesn't satisfy any one of them, the signature will be invalid, else the verifier will compute $(x_1', y_1')=sG+(r+s)P$.

Step 2. The verifier computes $r'=(h(m)+x_1') \bmod q$, then validates whether $r'=r$ or not. If $r'=r$, the signature will be valid, else the signature will be invalid.

3. Several Secret Sharing Algorithms

According to the signature process of SM2 elliptic curve digital signature algorithm in section 2, the signer chooses a random number k, then computes $kG=(x_1, y_1)$ in Step 1, so the secret sharing algorithm of a random secret number and the secret sharing algorithm of the multiplication of a number and a point are required in the design of the SM2 elliptic curve threshold signature scheme. The signer needs to compute $(1+d)^{-1}$, $k-rd$ and $(1+d)^{-1}(k-rd)$ in Step 3, so the secret sharing algorithm of the inverse of the secret, the secret sharing algorithm of the sum of the secrets or the difference between the secrets and the secret sharing algorithm of the multiplication of two secrets are required in the design of the SM2 elliptic curve threshold signature scheme. The secret sharing algorithm of 0 is used to improve the safety of the algorithm in the secret sharing algorithm of the multiplication of two secrets. Therefore, the above 6 secret sharing algorithms used in the design of the SM2 elliptic curve threshold signature scheme are introduced in this section.

3.1 Joint-Shamir-RSS [5]

In the joint Shamir random secret sharing algorithm, the participants U_1, U_2, $\cdots$, U_n chooses a random secret number respectively, and each participant distributes the secret shares of its

random secret number to the other participants. The special trusted center is not needed in this process. The participants U_1, U_2, $\cdots$, U_n realize the sharing of a random secret number which equals the sum of all the random secret numbers chosen by the participants. The detailed process of the joint Shamir random secret sharing algorithm is described as follows.

Step 1. Each participant U_i （$i=1,2,\cdots,n$） chooses a random secret number $a_o^{(i)}$ and constructs a secret polynomial $f_i(x)=\sum_{j=0}^{t}a_j^{(i)}x^j$ (where the order of $f_i(x)$ is t).

Step 2. Each participant U_i computes $f_i(g)$, $g=1,2,\cdots,n$, then sends $f_i(g)$ to the participant U_g secretly.

Step 3. The participant U_g receives the secret shares $f_i(g)$ sent by the other participants U_i （$1\le i\le n$ and $i\ne g$）, then computes $\sigma_g=\sum_{i=1}^{n}f_i(g)\bmod q$ as its secret share.

The above algorithm is called joint Shamir random secret sharing algorithm or Joint-Shamir-RSS algorithm for short. The participants U_1, U_2, $\cdots$, U_n share the secret $\sigma=\sum_{i=1}^{n}a_o^{(i)}$ by this algorithm. Accordingly, the polynomial used to share the secret σ is $f(x)=\sum_{j=0}^{t}a_jx^j$, where $a_j=\sum_{i=1}^{n}a_j^{(i)}$.

3.2 Joint-Shamir-ZSS [5]

The joint Shamir zero secret sharing algorithm is similar to the joint Shamir random secret sharing algorithm. The only difference is that the random secret number chosen by each participant U_i （$i=1,2,\cdots,n$） is $a_o^{(i)}=0$, so the secret shared by the participants is $\sigma=\sum_{i=1}^{n}a_o^{(i)}=0$. The joint Shamir zero secret sharing algorithm is called Joint-Shamir-ZSS algorithm for short.

3.3 Sum or Diff-SS

Suppose that the participant U_i gets the secret shares $u_i=f_u(i)$ and $v_i=f_v(i)$ of the secret numbers u and v respectively by Joint-Shamir-RSS algorithm, where $u=f_u(0)$, $v=f_v(0)$, $f_u(x)$ and $f_v(x)$ are two different polynomials whose order is t. Q is the set composed of the subscripts w of any $t+1$ participants' symbols U_w. The sum of the secret numbers u and v or the difference between the secret numbers u and v can be expressed as follows.

$$\begin{aligned} z&=u\pm v\\ &=\sum_{i\in Q}(u_i\prod_{j\in Q,j\ne i}\frac{j}{j-i})\pm\sum_{i\in Q}(v_i\prod_{j\in Q,j\ne i}\frac{j}{j-i})\\ &=\sum_{i\in Q}[(u_i\pm v_i)\prod_{j\in Q,j\ne i}\frac{j}{j-i}] \end{aligned} \tag{1}$$

Let z_i denote the participant's secret share of the secret number z which is got by Joint-Shamir-RSS algorithm, then

$$z = \sum_{i \in Q} (z_i \prod_{j \in Q, j \neq i} \frac{j}{j-i}) \quad (2)$$

According to equation (1) and equation (2), it can be derived that

$$z_i = u_i \pm v_i \quad (3)$$

That is to say, the participant's secret share of the sum of two secret numbers equals to the sum of the secret shares of the two secret numbers, and the participant's secret share of the difference between two secret numbers equals to the difference between the secret shares of the two secret numbers. The polynomial used to share the secret number z is $f_z(x) = f_u(x) \pm f_v(x)$. The order of the polynomial $f_z(x)$ is also t. The above algorithm is called sum of secrets or difference between secrets sharing algorithm, and it is called Sum or Diff-SS algorithm for short.

3.4 Mul-SS [5]

Suppose that the participant U_i gets the secret shares $u_i = f_u(i)$ and $v_i = f_v(i)$ of the secret numbers u and v respectively by Joint-Shamir-RSS algorithm, where $u = f_u(0)$, $v = f_v(0)$, $f_u(x)$ and $f_v(x)$ are two different polynomials whose order is t. Like the sum or diff-SS algorithm, the participant's secret share of the multiplication $h = uv$ of two secret numbers u and v is computed as follows.

$$h_i = u_i v_i \quad (4)$$

The polynomial used to share $h = uv$ is $f_h(x) = f_u(x) \cdot f_v(x)$. The order of the polynomial $f_h(x)$ is $2t$. So at least $2t+1$ participants can recover the secret $h = uv$. Since $f_h(x)$ is the multiplication of two polynomials, it is an irreducible polynomial and its coefficients are not random completely, which reduces the security. So in order to make the coefficients of $f_h(x)$ random and improve the security of $f_h(x)$, it is required to add a random polynomial whose order is $2t$ to $f_h(x)$. The detailed process is described as follows.

Step 1. Each participant U_i gets the secret share α_i of 0 by the Joint-Shamir-ZSS algorithm.

Step 2. Each participant U_i computes $h_i = u_i v_i + \alpha_i$ as the secret share of $h = uv$.

The order of the polynomial $f_h(x)$ is $2t$, so at least $2t+1$ participants can recover the secret $h = uv$. The above algorithm is called multiplication of secrets sharing algorithm or Mul-SS algorithm for short.

3.5 Inv-SS [5]

Suppose that the participants have shared the secret number u and U_i has got the secret share u_i of the secret number u. The detailed process of sharing u^{-1} by the participants is described as follows.

Step 1. The participants share the random secret number β by the Joint-Shamir-RSS algorithm. U_i gets the secret share β_i of the random secret number β.

Step 2. At least $2t+1$ participants share the multiplication βu of u and β by the Mul-SS algorithm. Each participant U_i gets the secret share $(\beta u)_i$ and broadcasts $(\beta u)_i$ to the other participants.

Step 3. Each participant U_i receives $(\beta u)_j$ （$1 \le j \le n$ and $j \neq i$） broadcasted by the other

participants, then computes βu as follows according to the interpolation formula.

$$\beta u = \sum_{i \in Q} [(\beta u)_i \prod_{j \in Q, j \neq i} \frac{j}{j-i}] \tag{5}$$

Where Q is the set composed of the subscripts w of any $2t+1$ participants' symbols U_w who share βu.

Step 4. Each participant U_i computes the secret share of u^{-1} as follows.

$$(u^{-1})_i = (\beta u)^{-1} \beta_i \bmod q \tag{6}$$

Since the order of the polynomial used to share the random secret number β is t, the order of the polynomial used to share u^{-1} is also t. So at least $t+1$ participants sharing u^{-1} can recover u^{-1}. The above algorithm is called inverse of secret sharing algorithm or Inv-SS algorithm for short.

3.6 PM-SS

Suppose that the participant U_i gets the secret share u_i of the secret number u by the Joint-Shamir-RSS algorithm. Q is the set composed of the subscripts w of any $t+1$ participants' symbols U_w. According to the interpolation formula, it can be derived that

$$\begin{aligned} uG &= [\sum_{i \in Q} (u_i \prod_{j \in Q, j \neq i} \frac{j}{j-i})]G \\ &= \sum_{i \in Q} [(u_i G) \prod_{j \in Q, j \neq i} \frac{j}{j-i})] \end{aligned} \tag{7}$$

So the participant U_i can get the secret share of the secret uG by computing $u_i G$, and at least $t+1$ participants can recover the secret uG. The above algorithm is called point multiplication secret sharing algorithm or PM-SS algorithm for short.

4. Design of the SM2 Elliptic Curve Threshold Signature Scheme

We can design a SM2 elliptic curve threshold signature scheme without a trusted center according to the above secret sharing algorithms. The scheme is divided into 3 parts including the initialization, the threshold signature and the signature validation.

4.1 Initialization

Suppose that the set of the participants is $\{U_1, U_2, \cdots, U_n\}$, $n \geq 2t+1$ and the message to be signed is m. The participants complete the initialization process before the threshold signature process. The participants share a random secret number by the Joint-Shamir-RSS algorithm. The random secret number is the private key d of the system actually. U_i gets the secret share d_i of the private key d, then computes $d_i G$ to get the secret share of dG and broadcasts $d_i G$. Each participant receives $d_i G$ broadcasted by the other participants, then computes the public key $P = dG$ of the system according to the equation (7). Besides, since $d' = (1+d)^{-1}$ is used in the signature process of the SM2 elliptic curve digital signature algorithm, the participants can complete the sharing of $d' = (1+d)^{-1}$ in the initialization process to improve the efficiency of the threshold signature scheme. According to the Inv-SS

algorithm, the detailed process of sharing $d' = (1+d)^{-1}$ by the participants is described as follows.

Step 1. The participants share the random secret number β by the Joint-Shamir-RSS algorithm. U_i gets the secret share β_i of the random secret number β.

Step 2. The participants choose the polynomials which have the order $2t$, then execute the Joint-Shamir-ZSS algorithm. U_i gets the secret share α_i of 0.

Step 3. According to the Mul-SS algorithm, U_i gets the secret share $\gamma_i = \beta_i(1+d_i) + \alpha_i$ of the multiplication $\gamma = \beta(1+d)$. Then U_i broadcasts γ_i to the other participants.

Step 4. Each participant U_i receives γ_j （$1 \le j \le n$ and $j \ne i$） broadcasted by the other participants, then computes γ as follows according to the interpolation formula.

$$\gamma = \beta(1+d) = \sum_{i \in Q} (\gamma_i \prod_{j \in Q, j \ne i} \frac{j}{j-i}) \tag{8}$$

Where Q is the set composed of the subscripts w of any $2t+1$ participants' symbols U_w who share $\gamma = \beta(1+d)$.

Step 5. Each participant U_i computes the secret share d_i' of $d' = (1+d)^{-1}$ as follows.

$$d_i' = \gamma^{-1}\beta_i \bmod q \tag{9}$$

4.2 Threshold Signature and Signature Validation

At least $2t+1$ participants complete the threshold signature process as follows.

Step 1. Since the signer chooses a random number k in the signature process of SM2 elliptic curve digital signature algorithm, the participants share the random secret number k by the Joint-Shamir-RSS algorithm in the threshold signature process. U_i gets the secret share k_i of the random secret number k.

Step 2. Since the participants need to get x_1 by computing $kG = (x_1, y_1)$ and each participant only has the secret share k_i of the random secret number k, the participants can compute kG by the PM-SS algorithm. According to the PM-SS algorithm, each participant U_i gets the secret share of kG by computing k_iG, then broadcasts k_iG. Each participant U_i receives the secret shares of kG broadcasted by the other participants, then computes

$$\begin{aligned} kG &= (x_1, y_1) \\ &= \sum_{i \in Q} [(k_iG) \prod_{j \in Q, j \ne i} \frac{j}{j-i})] \end{aligned} \tag{10}$$

Where Q is the set composed of the subscripts w of any $t+1$ participants' symbols U_w.

Step 3. Each participant U_i computes

$$r = (h(m) + x_1) \bmod q \tag{11}$$

Step 4. Since the signature $s = (1+d)^{-1}(k - rd) \bmod q$, the participants need to get the secret share of $k - rd$ by the Sum or Diff-SS algorithm to get the secret share of the signature s finally. According to the Sum or Diff-SS algorithm, each participant U_i computes the secret share $k_i - rd_i$ of $k - rd$.

Step 5. Since the signature $s = (1+d)^{-1}(k - rd) \bmod q$ and the participants have already got the secret shares of $(1+d)^{-1}$ and $k - rd$, the participants can get the secret share of the

signature s by the Mul-SS algorithm. According to the Mul-SS algorithm, the participants choose the polynomials which have the order $2t$, then execute the Joint-Shamir-ZSS algorithm. U_i gets the secret share μ_i of 0, then computes the secret share s_i of the signature $s=(1+d)^{-1}(k-rd) \bmod q$ as follows.

$$s_i=(d_i'(k_i-rd_i)+\mu_i) \bmod q \tag{12}$$

Step 6. Each participant U_i broadcasts his signature share s_i. When the number of the signature shares got by any one participant or signature receiver reaches $2t+1$, he can compute the signature s according to the interpolation formula. Thus the signature (r,s) of the message m is got.

The signature validation process of SM2 elliptic curve threshold signature scheme is the same as that of SM2 elliptic curve digital signature algorithm in section 2.

5. Correctness, Security and Efficiency Analysis

5.1 Correctness Analysis

Theorem 1. For the message m and the signature (r,s), if $r'=r$, then (r,s) will be the effective threshold signature of the message m.

Proof. According to the equations (8), (9), (12), we obtain

$$\begin{aligned} s_i &= (d_i'(k_i-rd_i)+\mu_i) \bmod q \\ &= ((\beta(1+d))^{-1}\beta_i(k_i-rd_i)+\mu_i) \bmod q \\ &= ((\beta(1+d))^{-1}\beta_i k_i-(\beta(1+d))^{-1}\beta_i rd_i+\mu_i) \bmod q \end{aligned}$$

Then according to the interpolation formula, we have

$$\begin{aligned} s &= \sum_{i\in Q}(s_i \prod_{j\in Q, j\neq i}\frac{j}{j-i}) \\ &= (\beta(1+d))^{-1}\sum_{i\in Q}(\beta_i k_i \prod_{j\in Q, j\neq i}\frac{j}{j-i})-(\beta(1+d))^{-1}r\sum_{i\in Q}(\beta_i d_i \prod_{j\in Q, j\neq i}\frac{j}{j-i})+\sum_{i\in Q}(\mu_i \prod_{j\in Q, j\neq i}\frac{j}{j-i}) \bmod q \\ &= (\beta(1+d))^{-1}\beta k-(\beta(1+d))^{-1}r\beta d+0 \bmod q \\ &= (1+d)^{-1}k-(1+d)^{-1}rd \bmod q \\ &= (1+d)^{-1}(k-rd) \bmod q \end{aligned}$$

and

$$\begin{aligned} (x_1', y_1') &= sG+(r+s)P \\ &= sG+(r+s)dG \\ &= (1+d)sG+rdG \\ &= (1+d)(1+d)^{-1}(k-rd)G+rdG \\ &= kG \\ &= (x_1, y_1) \end{aligned}$$

Thus $x_1'=x_1$

Consequently,

$$\begin{aligned} r' &= (h(m)+x_1') \bmod q \\ &= (h(m)+x_1) \bmod q \\ &= r \end{aligned}$$

Now proof is completed.

5.2 Security Analysis

The security of the threshold signature scheme in this paper is mainly based on the computational intractability of the Elliptic Curve Discrete Logarithm Problem (ECDLP) over a finite field. The definition of the Elliptic Curve Discrete Logarithm Problem is given as below.

Definition 1 (Elliptic Curve Discrete Logarithm Problem). Given the elliptic curve E defined over a finite field $GF(p)$, P and Q are any two points on the elliptic curve E. For a integer k, $Q = kP$. The problem that finding k according to P, Q and the elliptic curve E is called the Elliptic Curve Discrete Logarithm Problem (ECDLP) over the finite field $GF(p)$.

Over the past decade, the ECDLP has attracted the attention of many mathematicians from all over the world. It has been proved that it is relatively easy to solve Q according to k and P, but it is difficult to solve k according to Q and P. Until now, the effective solving method whose taking time is less than the exponential time has not been proposed for the general ECDLP. Therefore, it is difficult to solve the ECDLP over a finite field, that is to say, it is infeasible on computation.

This scheme may suffer from the attacks of the insiders or the external attackers in the threshold signature process and the signature validation process. It can be shown that this scheme can resist the common attacks by the following analysis.

Attack 1. The attacker tends to solve the private key of the system using the open information and the signature.

Situation 1. The attacker tends to solve the private key d of the system using the public key P of the system.

Since $P = dG$, it is equivalent to solving the Elliptic Curve Discrete Logarithm Problem (ECDLP) that the attacker tends to solve d using P, which is difficult on computation.

Situation 2. The attacker tends to solve the private key d of the system using the message m and its threshold signature (r, s).

According to the proof process of Theorem 1, the attacker can compute $kG = sG + (r+s)P$, but it is equivalent to solving the Elliptic Curve Discrete Logarithm Problem (ECDLP) that solving the random number k on this basis, which is difficult on computation. Therefore two unknown numbers d and k exist in the equation $s = (1+d)^{-1}(k - rd) \bmod q$ for the attacker. It is impossible to solve the private key d of the system.

Attack 2. The attacker tends to solve the private key share d_i of the participant U_i using the signature share s_i of U_i $(i = 1, 2, \cdots, n)$.

According to the proof process of Theorem 1, many unknown numbers β, d, β_i, k_i and μ_i exist in the equation $s_i = ((\beta(1+d))^{-1}\beta_i k_i - (\beta(1+d))^{-1}\beta_i r d_i + \mu_i) \bmod q$ for the attacker. Therefore it is impossible to solve the private key share d_i of the participant U_i using the signature share s_i of U_i $(i = 1, 2, \cdots, n)$.

Attack 3. The attacker tends to forge the threshold signatures of other messages using the open information in the threshold signature scheme.

Situation 1. The attacker tends to forge the signature share s_i of the message m produced by the participant U_i according to the threshold signature scheme, then use $2t+1$ signature shares to forge the threshold signature s of the message m according to the interpolation formula.

According to the proof process of Theorem 1, the signature share of the message m

produced by the participant U_i is $s_i = ((\beta(1+d))^{-1}\beta_i k_i - (\beta(1+d))^{-1}\beta_i r d_i + \mu_i) \bmod q$. Since the attacker doesn't know the private key d of the system or the private key share d_i of the participant U_i, he can't forge the signature share s_i of the message m produced by the participant U_i. Consequently, the attacker can't forge the threshold signature s of the message m according to the interpolation formula.

Situation 2. The attacker tends to forge the threshold signature (r,s) of the message m directly by making them satisfy the signature validation equations $(x_1', y_1') = sG + (r+s)P$ and $(h(m) + x_1') \bmod q = r$.

If the attacker chooses the message m and a random number k, he can compute $(x_1', y_1') = kG$ and $r = (h(m) + x_1') \bmod q$. But it is not easier than solving the Elliptic Curve Discrete Logarithm Problem (ECDLP) that solving s according to the equation $(x_1', y_1') = sG + (r+s)P$, which is difficult on computation.

If the attacker chooses the message m and r, he can solve x_1' according to the equation $r = (h(m) + x_1') \bmod q$. But it is not easier than solving the Elliptic Curve Discrete Logarithm Problem (ECDLP) that solving s according to the equation $(x_1', y_1') = sG + (r+s)P$, which is difficult on computation.

If the attacker chooses the message m and s, it will not be easier than solving the Elliptic Curve Discrete Logarithm Problem (ECDLP) that solving r according to the signature validation equations $(x_1', y_1') = sG + (r+s)P$ and $(h(m) + x_1') \bmod q = r$, which is difficult on computation.

5.3 Efficiency Analysis

The efficiency of the proposed SM2 elliptic curve threshold signature scheme is analyzed from the traffic and the calculation in this paper. And the proposed SM2 elliptic curve threshold signature scheme is compared with the threshold signature scheme based on Elliptic Curve Digital Signature Algorithm (ECDSA) in efficiency.

We assume that the parameters of the ECDSA are the same as that of the SM2 elliptic curve public key algorithms, therefore the signature process of ECDSA can be described as follows.

Step 1. The signer chooses a random number $k \in [1, q-1]$, then computes $kG = (x_1, y_1)$.

Step 2. The signer computes $r = x_1 \bmod q$. If $r = 0$, the signer will go back to Step 1.

Step 3. For the message m to be signed, the signer computes $s = k^{-1}(h(m) + rd) \bmod q$. If $s = 0$, the signer will go back to Step 1, else he will get the signature (r,s) of the message m.

The signature validation process of ECDSA is described as follows.

Step 1. When the verifier receives the message m and its signature (r,s), he will validate whether (r,s) satisfy $r, s \in [1, q-1]$ or not. If (r,s) doesn't satisfy $r, s \in [1, q-1]$, the signature will be invalid, else the verifier will compute $w = s^{-1} \bmod q$.

Step 2. The verifier computes $u_1 = h(m)w \bmod q$ and $u_2 = rw \bmod q$.

Step 3. The verifier computes $(x_1', y_1') = u_1 G + u_2 G$ and $r' = x_1' \bmod q$, then validates whether $r' = r$ or not. If $r' = r$, the signature will be valid, else the signature will be invalid.

Like the design of the SM2 elliptic curve threshold signature scheme, the secret sharing algorithms introduced in section 3 are still used to design the threshold signature scheme based on ECDSA. The participant U_i gets the secret share d_i of the private key d and the public key $P = dG$ of the system in the initialization process of the threshold signature scheme. Then, at

least $2t+1$ participants complete the threshold signature process as follows.

Step 1. The participants share the random secret number k by the Joint-Shamir-RSS algorithm. U_i gets the secret share k_i of the random secret number k.

Step 2. According to the PM-SS algorithm, each participant U_i gets the secret share of kG by computing k_iG, then broadcasts k_iG. Each participant U_i receives the secret shares of kG broadcasted by the other participants, then computes

$$\begin{aligned} kG &= (x_1, y_1) \\ &= \sum_{i \in Q} [(k_i G) \prod_{j \in Q, j \neq i} \frac{j}{j-i})] \end{aligned} \tag{13}$$

Where Q is the set composed of the subscripts w of any $t+1$ participants' symbols U_w.

Step 3. Each participant U_i computes

$$r = x_1 \bmod q \tag{14}$$

Step 4. Each participant U_i computes the secret share $h(m)+rd_i$ of $h(m)+rd$.

Step 5. The participants share k^{-1} by the Inv-SS algorithm. U_i gets the secret share k_i' of k^{-1}.

Step 6. The participants share the signature $s = k^{-1}(h(m)+rd) \bmod q$ by the Mul-SS algorithm. U_i gets the secret share s_i of s.

Step 7. Each participant U_i broadcasts his signature share s_i. When the number of the signature shares got by any one participant or signature receiver reaches $2t+1$, he can compute the signature s according to the interpolation formula. Thus the signature (r, s) of the message m is got.

The signature validation process of the threshold signature scheme based on ECDSA is the same as that of the ECDSA above.

According to the above threshold signature scheme, it can be seen that the main differences between the proposed SM2 elliptic curve threshold signature scheme and the threshold signature scheme based on ECDSA include the following two points. First, the participants need not to execute the Inv-SS algorithm in the threshold signature process of the SM2 elliptic curve threshold signature scheme, but the participants need to execute the Inv-SS algorithm to share k^{-1} in the threshold signature process of the threshold signature scheme based on ECDSA. Second, the verifier needs not to execute the inverse computation and the modular multiplication computation in the signature validation process of the SM2 elliptic curve threshold signature scheme, but the verifier needs to execute the inverse computation to compute $w = s^{-1} \bmod q$ and execute the modular multiplication to compute $u_1 = h(m)w \bmod q$ and $u_2 = rw \bmod q$ in the signature validation process of the threshold signature scheme based on ECDSA.

5.3.1 Traffic Analysis

The traffic of the threshold signature scheme is estimated by the amount of the data transmitted in the scheme in this paper. $|x|$ denotes the data length of the number x in the following analysis, and its unit is bit.

Since the initialization process of the threshold signature scheme has been completed before the threshold signature process, and the verifier needs not to communicate with others in the signature validation process, only the traffic in the threshold signature process is considered.

Suppose that the number of the participants is T $(T \geq 2t+1)$ in the SM2 elliptic curve threshold signature scheme. Each participant needs to transmit $(T-1)|q|$ bits of data to the other $T-1$ participants secretly when executing the Joint-Shamir-RSS algorithm to share the random secret number k. Each participant needs to broadcast $2|p|$ bits of data when broadcasting k_iG. Each participant needs to transmit $(T-1)|q|$ bits of data to the other $T-1$ participants secretly when executing the Joint-Shamir-ZSS algorithm. Each participant needs to broadcast $|q|$ bits of data when broadcasting his signature share s_i. Therefore each participant needs to transmit a total of $2(T-1)|q|$ bits of data secretly and broadcast a total of $2|p|+|q|$ bits of data in the threshold signature process.

Suppose that the number of the participants is T $(T \geq 2t+1)$ in the threshold signature scheme based on ECDSA designed by the same secret sharing algorithms. Each participant needs to transmit $(T-1)|q|$ bits of data to the other $T-1$ participants secretly when executing the Joint-Shamir-RSS algorithm to share the random secret number k and broadcast $2|p|$ bits of data when broadcasting k_iG. In the process of sharing k^{-1} by the participants, according to the Inv-SS algorithm, each participant needs to transmit $(T-1)|q|$ bits of data to the other $T-1$ participants secretly when executing the Joint-Shamir-RSS algorithm to share the random secret number β, transmit $(T-1)|q|$ bits of data to the other $T-1$ participants secretly when executing the Joint-Shamir-ZSS algorithm, and broadcast $|q|$ bits of data when executing the Mul-SS algorithm. After this, the participants share $s=k^{-1}(h(m)+rd) \bmod q$ by the Mul-SS algorithm. Each participant needs to transmit $(T-1)|q|$ bits of data to the other $T-1$ participants secretly when executing the Joint-Shamir-ZSS algorithm. At last, each participant needs to broadcast $|q|$ bits of data when broadcasting his signature share s_i. Therefore each participant needs to transmit a total of $4(T-1)|q|$ bits of data secretly and broadcast a total of $2|p|+2|q|$ bits of data in the threshold signature scheme based on ECDSA designed by the secret sharing algorithms in the SM2 elliptic curve threshold signature scheme.

Since the number of the participants is T, the traffic contrast between the two threshold signature schemes is shown in **Table 1**. Obviously the traffic of the SM2 elliptic curve threshold signature scheme is less than that of the threshold signature scheme based on ECDSA. The main reason is that the process of sharing the inverse of the secret has been completed in the initialization of the SM2 elliptic curve threshold signature scheme and is not needed in the threshold signature process, but the process of sharing k^{-1} needs to be completed and increases the traffic to some extent in the threshold signature process of the threshold signature scheme based on ECDSA.

Table 1. Traffic contrast between the two threshold signature schemes

Threshold signature scheme	Data transmitted secretly (bit)	Data broadcasted (bit)
SM2 elliptic curve threshold signature scheme	$2T(T-1)\|q\|$	$T(2\|p\|+\|q\|)$
Threshold signature scheme based on ECDSA	$4T(T-1)\|q\|$	$T(2\|p\|+2\|q\|)$

5.3.2 Calculation Analysis

Compared with the hash computation, the point addition computation and the scalar multiplication computation based on the elliptic curve E, the inverse computation, the exponent computation and the modular multiplication computation over the finite field Z_q, the calculated amount of the other computations is smaller in the threshold signature scheme, so the number of the above computations are used to estimate the calculated amount of the threshold signature scheme in this paper.

The calculation analysis is similar to the traffic analysis. Since the initialization process of the threshold signature scheme has been completed before the threshold signature process, only the calculation in the threshold signature process and the signature validation process is considered. Suppose that the number of the participants is T $(T \geq 2t+1)$ in the threshold signature process of the SM2 elliptic curve threshold signature scheme. Each participant needs to execute $T \cdot t$ exponent computations when executing the Joint-Shamir-RSS algorithm to share the random secret number k. According to the PM-SS algorithm, each participant needs to execute $t+2$ scalar multiplication computations and t point addition computations in the process of getting kG. Each participant needs to execute 1 hash computation when computing $r = (h(m) + x_1) \bmod q$. Each participant needs to execute $2T \cdot t$ exponent computations when executing the Joint-Shamir-ZSS algorithm to share 0 in the process of getting the secret share s_i of s according to the Mul-SS algorithm. In the signature validation process of the SM2 elliptic curve threshold signature scheme, the verifier needs to execute 2 scalar multiplication computations and 1 point addition computation when computing $(x_1', y_1') = sG + (r+s)P$ and execute 1 hash computation when computing $r' = (h(m) + x_1') \bmod q$. One scalar multiplication computation is composed of a number of point addition computations. One scalar multiplication computation mG can be transformed into $2(|q|-1)$ point addition computations for the integer $m \in Z_q$ and the base point G on the elliptic curve. Therefore the calculation of each participant can be equivalent to $3T \cdot t$ exponent computations, $2(t+2)(|q|-1)+t$ point addition computations and 1 hash computation in the threshold signature process of the SM2 elliptic curve threshold signature scheme, and the calculation of the verifier can be equivalent to $4(|q|-1)+1$ point addition computations and 1 hash computation in the signature validation process of the SM2 elliptic curve threshold signature scheme.

Suppose that the number of the participants is T $(T \geq 2t+1)$ in the threshold signature process of the threshold signature scheme based on ECDSA designed by the same secret sharing algorithms. Each participant needs to execute $T \cdot t$ exponent computations when executing the Joint-Shamir-RSS algorithm to share the random secret number k. According to the PM-SS algorithm, each participant needs to execute $t+2$ scalar multiplication computations and t point addition computations in the process of getting kG. Each participant needs to execute 1 hash computation when computing $h(m) + rd_i$. In the process of sharing k^{-1} by the participants, according to the Inv-SS algorithm, each participant needs to execute $T \cdot t$ exponent computations when executing the Joint-Shamir-RSS algorithm to share the random secret number β, execute $2T \cdot t$ exponent computations when executing the Joint-Shamir-ZSS algorithm to share 0, and execute 1 inverse computation and 1 modular

multiplication computation when computing k_i'. In the process of getting the secret share s_i of s by the Mul-SS algorithm, each participant needs to execute $2T \cdot t$ exponent computations when executing the Joint-Shamir-ZSS algorithm to share 0. In the signature validation process of the threshold signature scheme based on ECDSA, the verifier needs to execute 1 inverse computation when computing $w = s^{-1} \bmod q$, execute 1 hash computation and 2 modular multiplication computations when computing $u_1 = h(m)w \bmod q$ and $u_2 = rw \bmod q$, and execute 2 scalar multiplication computations and 1 point addition computation when computing $(x_1', y_1') = u_1 G + u_2 G$. Since one scalar multiplication computation can be transformed into $2(|q|-1)$ point addition computations, the calculation of each participant can be equivalent to $6T \cdot t$ exponent computations, $2(t+2)(|q|-1)+t$ point addition computations, 1 hash computation, 1 inverse computation and 1 modular multiplication computation in the threshold signature process of the threshold signature scheme based on ECDSA, and the calculation of the verifier can be equivalent to 1 inverse computation, 1 hash computation, 2 modular multiplication computations and $4(|q|-1)+1$ point addition computations in the signature validation process of the threshold signature scheme based on ECDSA.

Let T_h, T_{pa}, T_i, T_e and T_{mm} be the time for performing a hash computation, a point addition computation, a inverse computation, a exponent computation and a modular multiplication computation respectively. Suppose that the number of the participants is T $(T \geq 2t+1)$. The calculation contrast between the threshold signature processes of the two threshold signature schemes is shown in **Table 2**, and the calculation contrast between the signature validation processes of the two threshold signature schemes is shown in **Table 3**.

Table 2. Calculation contrast between the threshold signature processes of the two threshold signature schemes

Threshold signature scheme	Calculation in the threshold signature process
SM2 elliptic curve threshold signature scheme	$3T^2 \cdot tT_e + T[2(t+2)(\|q\|-1)+t]T_{pa} + TT_h$
Threshold signature scheme based on ECDSA	$6T^2 \cdot tT_e + T[2(t+2)(\|q\|-1)+t]T_{pa} + TT_h + TT_i + TT_{mm}$

Table 3. Calculation contrast between the signature validation processes of the two threshold signature schemes

Threshold signature scheme	Calculation in the signature validation process
SM2 elliptic curve threshold signature scheme	$[4(\|q\|-1)+1]T_{pa} + T_h$
Threshold signature scheme based on ECDSA	$[4(\|q\|-1)+1]T_{pa} + T_h + T_i + 2T_{mm}$

Obviously the calculation in the threshold signature process and the signature validation process of the SM2 elliptic curve threshold signature scheme is less than that in the threshold signature process and the signature validation process of the threshold signature scheme based on ECDSA respectively. The main reasons include the following two points. First, the process of sharing the inverse of the secret has been completed in the initialization of the SM2 elliptic curve threshold signature scheme and is not needed in the threshold signature process, but the process of sharing k^{-1} needs to be completed in the threshold signature process of the threshold signature scheme based on ECDSA, which causes the participants to execute the

Inv-SS algorithm and increases the calculation to some extent. Second, the verifier needs not to execute the inverse computation and modular multiplication in the signature validation process of the SM2 elliptic curve threshold signature scheme, but the verifier needs to execute the inverse computation and the modular multiplication computation when computing w, u_1 and u_2 in the signature validation process of the threshold signature scheme based on ECDSA, which increases the calculation to some extent.

6. Conclusion

A SM2 elliptic curve threshold signature scheme without a trusted center based on the SM2 elliptic curve digital signature algorithm released by Chinese Encryption Administration in December 2010 is proposed according to several secret sharing algorithms in this paper. The proposed scheme is analyzed from correctness, security and efficiency. The correctness analysis shows that the proposed scheme can realize the effective threshold signature. The security analysis shows that the proposed scheme can resist some kinds of common attacks. The efficiency of the proposed scheme is analyzed from the traffic and the calculation. If the secret sharing algorithms in this paper are used to design the threshold signature schemes, the traffic and the calculation of the designed SM2 elliptic curve threshold signature scheme will be all less than that of the designed threshold signature scheme based on ECDSA, that is to say, the former will be more efficient than the latter. At present, the papers on the threshold signature scheme based on the SM2 elliptic curve digital signature algorithm are few. The number of the signers may depend on the importance of the message to be signed in many practical applications, which demands that the threshold value of the threshold signature scheme be dynamic. Therefore the threshold signature scheme with dynamic threshold value based on the SM2 elliptic curve digital signature algorithm will be the next important research content.

References

[1] BOYD C, "Digital multisignatures," in *Proc. of Cryptography and Coding*, pp. 241-246, December 17-21, 1986. Article (CrossRef Link).

[2] Desmedt Y and Frankel Y, "Threshold cryptosystems," in *Proc. of Advances in Cryptology*, pp. 307–315, August 20–24, 1989. Article (CrossRef Link).

[3] Desmedt Y and Frankel Y, "Shared generation of authenticators and signatures," in *Proc. of Advances in Cryptology*, pp. 457–469, August 11-15, 1991. Article (CrossRef Link).

[4] Harn L, "Group-oriented (t,n) threshold digital signature scheme and digital multisignature," *IEE Proceedings-Computers and Digital Techniques*, vol. 141, no. 5, pp. 307-313, September, 1994. Article (CrossRef Link).

[5] Gennaro R, Jarecki S and Krawczyk H, "Robust threshold DSS signatures," in *Proc. of Advances in Cryptology*, pp. 354–371, May 12–16, 1996. Article (CrossRef Link).

[6] Wang C T, Lin C H and Chang C C, "Threshold signature schemes with traceable signers in group communications," *Computer Communications*, vol. 21, no. 8, pp. 771–776, June, 1998. Article (CrossRef Link).

[7] Miyazaki K and Takaragi K, "A threshold digital signature scheme for a smart card based system," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 84, no. 1, pp. 205–213, January, 2001. Article (CrossRef Link).

[8] HAN Jin-rong, LU Ji-qiang and WANG Xin- mei, "A verifiable threshold signature scheme based on the elliptic curve," *Journal of Xidian University*, vol. 30, no. 1, pp. 26–28, February, 2003. Article (CrossRef Link).

[9] Peng Hua-xi and Feng Deng-guo, "A forward secure threshold signature scheme from bilinear pairing," *Journal of Computer Research and Development*, vol. 44, no. 4, pp. 574–580, April, 2007. Article (CrossRef Link).
[10] SHEN Zhong-hua and YU Xiu-yuan, "Threshold signature scheme with threshold verification based on multivariate linear polynomial," *Journal of Shanghai Jiaotong University (Science)*, vol. 16, no. 5, pp. 551–556, October, 2011. Article (CrossRef Link).
[11] Slim Bettaieb and Julien Schrek, "Improved lattice-based threshold ring signature scheme," *Lecture Notes in Computer Science*, vol. 7932, no. 1, pp. 34-51, June, 2013. Article (CrossRef Link).
[12] Qingbin Wang, Shaozhen Chen and Aijun Ge, "A new lattice-based threshold attribute-based signature scheme," *Lecture Notes in Computer Science*, vol. 9065, no. 1, pp. 406-420, May, 2015. Article (CrossRef Link).
[13] Fei Li, Wei Gao, Guilin Wang, Kefei Chen and Xueli Wang, "Efficient identity-based threshold signature scheme from bilinear pairings in standard model," *International Journal of Internet Protocol Technology*, vol. 8, no. 2-3, pp. 107-115, January, 2014. Article (CrossRef Link).
[14] Jung Yeon Hwang, Hyoung Joong Kim, Dong Hoon Lee and Boyeon Song, "An enhanced (t, n) threshold directed signature scheme," *Information Sciences*, vol. 275, pp. 284-292, August, 2014. Article (CrossRef Link).
[15] Raman Kumar, Harsh Kumar Verma and Renu Dhir, "Analysis and design of protocol for enhanced threshold proxy signature scheme based on RSA for known signers," *Wireless Personal Communications*, vol. 80, no. 3, pp. 1281-1345, February, 2015. Article (CrossRef Link).
[16] Chinese Encryption Administration, "SM2 elliptic curve public key algorithms," December, 2010. Article (CrossRef Link).

Yan Jie was born in Hanzhong, Shaanxi Province, China, in 1988. He is now a Ph.D. candidate in Equipment Command and Management Department, Mechanical Engineering College, Shijiazhuang, China. He received the B.S. degree from China University of Petroleum, Dongying, China in 2010 and the M.S. degree from Mechanical Engineering College, Shijiazhuang, China in 2012. His research interests include network security, communication system and cryptograph.

Lu Yu was born in Luoyang, Henan Province, China, in 1960. He is now a doctor, professor, doctoral tutor in Equipment Command and Management Department, Mechanical Engineering College, Shijiazhuang, China. He received the M.S. degree from National University of Defence Technology, Changsha, China in 1990 and the Ph.D. degree from Beijing University of Aeronautics and Astronautics, Beijing, China in 2001. His research interests include information security, computer network and cryptograph.

Chen Li-yun was born in Anren, Hunan Province, China, in 1969. He is now a doctor, professor, master's supervisor in Information Engineering Department,, Mechanical Engineering College, Shijiazhuang, China. He received the M.S. degree from Nanjing University of Science and Technology, Nanjing, China in 1990 and the Ph.D. degree from Mechanical Engineering College, Shijiazhuang, China in 2014. His research interests include information security, computer network and cryptograph.

Nie Wei was born in Sanmenxia, Henan Province, China, in 1973. He received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China in 2011. He was a Postdoctoral Fellow with the Department of Computer Science, Yale University, New Haven, CT from 2011 to 2013. He is currently a Lecturer with the College of Information Engineering, Shenzhen University, Shenzhen, China. His research interests include network security, communication system and optimization theory.