JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 37, 1449-1466 (2021)
DOI: 10.6688/JISE.202111_37(6).0015

A New Attack for Self-Certified Digital Signatures for E-Commerce Applications

CHIN-YU SUN[1], HSIAO-LING WU[2], HUNG-MIN SUN[1] AND TINGTING HWANG[1]
[1]*Department of Computer Science*
National Tsing Hua University
Hsinchu, 300 Taiwan
E-mail: sun.chin.yu@gmail.com; {hmsun; tingting}@cs.nthu.edu.tw
[2]*Department of Information Management*
Chaoyang University of Technology
Taichung, 413 Taiwan
E-mail: wuhsiaoling590@gmail.com

"Self-certified digital signature with message recovery" allows a specific receiver to restore the meaningful message from a digital signature and simultaneously confirms the validity of a signature and a signer's public key. This method greatly improves message confidentiality, solves the certificate management problem, and reduces the communication costs. Due to those benefits, this signature scheme has been widely adopted for *e*-commerce applications. However, in recent years, this method has attracted attackers' attention; hence, a series of schemes were proposed to counter different attack scenarios. In this paper, we will first present a new attack scenario that can break the security of all the "self-certified digital signature with message recovery" schemes. Then, we will propose a scheme to solve the security issues. Compared with this type of signature scheme, our scheme can satisfy the essential security requirement of a digital signature without sacrificing the cost-effectiveness of the original design. The security and performance analyses demonstrate that our proposed scheme is secure, efficient, and well suited for practical use in *e*-commerce.

Keywords: digital signature, self-certified, message recovery, provable security, E-commerce

1. INTRODUCTION

A digital signature is one realization of public key cryptosystems. A set of keys is divided into a public key and a private key. As long as an owner keeps the private key, other people can easily use the corresponding public key to reaffirm the subtle relationship between the public and private keys without revealing any private key information. More precisely, a signer cannot deny his signature once he signed a message by his private key; further, a receiver, also called a verifier, can confirm the source of the signature via the signer's public key. Hence, the receiver can notarize a digital document, such as a contract. This scheme has already been widely applied in *e*-commerce, *e.g.*, *e*-auction, online transactions, *etc.*

To ensure the identity of a signer, the digital signature schemes require a trusted third-party to issue a certificate for linking the signer with his/her public key. Therefore, a verifier can identify the source of a signature with the help of a certificate. Unfortunately, a trusted third-party requires a substantial amount of storage space and computation costs to maintain the certificates. Hence, a series of "self-certified digital signature with message

Received November 11, 2019; revised April 4, 2020; accepted June 23, 2020.
Communicated by Fu-Hau Hsu.

recovery" schemes were proposed to overcome the above issues.

In 1991, Girault [1] first introduced the notion of a self-certified public key with a certificate embedded in the public key instead of attached with a public key. The identity linkage between a user and his/her public key thus becomes a strong relationship. Further, Girault's signature scheme no longer relies on a certificate; hence, the trusted third-party can reduce the cost from setting up and maintaining the storage space. Moreover, a great benefit of the self-certified technique is that the signer is no longer needed to send the certificate to the verifier; furthermore, the verifier is also no longer needed to communicate with the trusted third-party to verify its certificate and the signer's public key. In 2003, Tseng *et al.* [2] extended Girault's concept and introduced the self-certified sig-nature scheme with message recovery. In this scheme, the message is hidden in the signature. Therefore, only a specific verifier can extract the information from the signature and verify it. Compared with the appendix-based digital signature [2], the message-recovery technique can protect the privacy of the message without extra encryption/decryption. Due to the benefits of self-certified and message-recovery techniques, this new type of signature scheme contributes to reducing the costs from the storage space, communication, and computation.

In 2004, Shao [3] pointed out that Tseng *et al.*'s scheme demonstrates three security weaknesses, *i.e.*, insider forgery attack, repudiation problem, and failure arbitration. Therefore, based on Tseng's scheme, Shao proposed an improved scheme to overcome these weaknesses. In 2005, Chang *et al.*'s [4] presented a digital signature scheme by preserving Tseng *et al.* scheme's properties without the assumption that the system authority is trustworthy. In addition, in 2006, Yoon and Yoo [5] and Zhang *et al.* [6] demonstrated that Chang *et al.*'s scheme inherits the known plaintext-ciphertext attack and message leakage problem, respectively.

In 2005, Lv *et al.* [7] demonstrated that, if only the designated receiver has ability to recover the message, then nobody except for him/herself can execute the verification algorithm. In such a case, no one can arbitrate the signature when a dishonest signer repudiates the signature. To solve this problem, Lv *et al.* designed a conversion phase for this situation. Subsequently, in 2006, Shao [8] revealed that Lv *et al.*'s scheme has a confidentiality issue. In 2009, Zhang *et al.* [9] further pointed out that Shao's scheme still suffers from the man-in-the-middle attack. In 2015, Wu and Xu [10] gave a complete analysis of the man-in-the-middle attack via the schemes [5, 8]. In addition, Wu and Xu proposed an improved scheme to overcome the weakness. Recently, Sadeghpour [11] further pointed out that Wu and Xu's scheme is not secure.

Because the digital signature is widely applied in *e*-commerce, we will investigate the security issue in *e*-commerce. We have discovered a new attack scenario that can break the security of all the "self-certified digital signature with message recovery" schemes. An attacker can easily impersonate a seller by forging a signature and obtaining payment from a buyer. To solve this problem, we develop a scheme that can satisfy the essential security requirement of a digital signature without sacrificing the cost-effectiveness of the original design.

The rest of the paper is organized as follows. Section 2 first introduces the basic definitions of our attacks and then gives the security requirements. Section 3 describes the attack scenario using Wu and Xu's scheme. Section 4 presents our proposed scheme. Then, we explain the difference between Wu and Xu's scheme and our scheme in Section 5. Section 6 gives a formal security analysis (under the random oracle model). Finally, Section 7 presents experimental results, with conclusions in Section 8.

2. BACKGROUND

In this section, we first described the entity and a simple transaction procedure in an *e*-ticket system. We then define two attack scenarios, based on this *e*-ticket system, for a further discussion in Section 3.

2.1 Adversarial Scenarios

Consider the following scenario, in a normal *e*-ticket system, when a buyer, Bob, wants to buy an *e*-ticket for a movie from an *e*-ticket store, Bob first sends a request message to the *e*-ticket store, and then the store generates a signature on an *e*-ticket and sends it to Bob. After receiving the message, Bob recovers the ticket and verifies both ticket and signature. When Bob uses it at a Cinema, the Cinema also needs to recover the ticket and verify it.

Type I Attack (Signer-oriented): In this attack, an attacker Eve, first targets a seller (*e.g.*, *e*-ticket store) and a digital signature (*e.g.*, an *e*-ticket). Then, the attacker has the ability to impersonate the seller to sell this ticket. Moreover, Bob cannot detect the attacker by the signature verification algorithm proposed in the previous literature. That is, the attacker can obtain the benefits (*e.g.*, the ticket's money) from the buyer. This attack totally breaks the security of a transaction agreement in an *e*-commerce system. Consequently, we can formally define our attack (Type I attack) as follows:

$$(pk_{attacker}, \sigma_{target}) \leftarrow \beta_1(pk_{attacker}, \sigma_{target}), \tag{1}$$

where $\beta_1(\cdot)$ is our Type I attack function, pk_{target} is the public key of a specific signer, σ_{target} is the signature of a specific signer, $pk_{attacker}$ is the public key of an attacker, and $\leftarrow$ denotes the output of the attack function. Type I attack in Eq. (1) means, by given a specific signer's public key pk_{target} and signature σ_{target}, the attacker can generate another pair $\{pk_{attacker}, \sigma_{target}\}$ and pass the signature verification algorithm. Then, we say that attacker successfully forges a signature by Type I attack.

Type II Attack (Signer-oriented): Sadeghpour [11] pointed out the other type of attack. In Sadeghpour's example, Bob and Charlie are friends and trust each other. After Bob buying an *e*-ticket from ticket store, he can generate another fake ticket by Charlie's private key; however, Charlie's signature is unauthorized but no one can prove it. We then defined the concept of the Type II attack as follows:

$$(pk_{user}, \sigma_{user}) \leftarrow \beta_2(pri_{user}, \sigma_{attacker}), \tag{2}$$

where $\beta_2(\cdot)$ is Type II attack function in [11], pri_{user} is the private key of a user, $\sigma_{attacker}$ is a signature of an attacker, pk_{user} is the public key of the user, and σ_{user} is the signature of the user. Type II attack in Eq. (2) means, by given an attacker's signature $\sigma_{attacker}$ and a private key of an unauthorized user pri_{user}, the attacker can generate another pair fake signature $\{pk_{user}, \sigma_{user}\}$ for the unauthorized user to pass the signature verification algorithm. Then, we say that the malicious receiver successfully forges a signature by Type II attack.

Here, we mentioned Type II attack because it is the newest security analysis for the same type of signature schemes. We will show our proposed scheme can also against Sadeghpour's attack in Section 5. We refer the reader to, *e.g.* [11], for a formal introduction to this attack.

2.2 Security Requirements

In this subsection, we provide formal security requirements for the new architecture of a self-certified digital signature with message recovery.

Correctness. A well-designed digital signature scheme should provide a verification algorithm that can help any third party to verify the signature. Once the signature is incorrect, even a little bit of error, the algorithm must output the error message to tell the verifier.

Unforgeable. A well-designed digital signature scheme should have the ability to detect the ownership of a digital signature. Once a malicious attacker tries to forge a signature, the signature verification algorithm should detect it and terminate the procedure immediately.

Non-repudiation. A well-designed digital signature scheme should have to link a digital signature with the signer strongly. Once a signer has signed a digital signature, he/she could not deny this behavior.

The up-to-date "self-certified digital signature with message recovery schemes" was proposed by Wu and Xu [10], which covers major aspects of the signature and provides a formal security proof. In the following, we will use their scheme as our base scheme for security analysis.

3. INSUFFICIENCY OF PREVIOUSLY PROPOSED SIGNATURES

The uptodate "self-certified digital signature with message recovery schemes" was proposed by Wu and Xu [10], which covers major aspects of the signature and provides a formal security proof. In the following, we will use their scheme as our base scheme for security analysis.

3.1 Review of Wu and Xu's Scheme

Wu and Xu's scheme includes four phases: system initialization phase, signature generation phase, message recovery phase, and signature conversation phase. The details of these phases are described below:

***System initialization phase*:** System Authority (SA) first chooses two large primes p and q such that $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are also primes. Then, SA computes $N' = p' \cdot q'$ to choose an integer g, which is a base element of order N'. After that, SA keeps p, q, p' and q' secret and publishes N and g to all users, where $N = p \cdot q$. SA also publishes several hash functions $F()$: $\{0, 1\}^* \rightarrow \{0, 1\}^k$, $F_1()$: $\{0, 1\}^* \rightarrow \{0, 1\}^{l_1}$, $F_2()$: $\{0, 1\}^* \rightarrow \{0, 1\}^{l_2}$, and $H()$: $\{0, 1\}^* \rightarrow \{0, 1\}^k$, where $k = \log(\min(p', q'))$ and $l_N = l_1 + l_2$. Once SA finishes the setup, it starts to serve users. When a user U_i wants to join this system, he first chooses $x_i \in_R Z_N^*$ and computes $p_i = g^{x_i} \bmod N$. Then, U_i sends his identity ID_i and p_i to SA. After receiving ID_i and p_i, SA responds U_i by giving U_i's public key $y_i = (p_i - ID_i)^{F^{-1}(ID_i)}$ mod N. Finally, U_i can compute $p_i' = y_i^{F(ID_i)} + ID_i$ and checks $p_i' = p_i$ to validate if this public key belongs to him.

***Signature generation phase*:** When U_i wants to sign a message $m \in \{0, 1\}$ to U^l, he firstly chooses $r_i \in_R Z_N^*$ and then computes $e_i = g^{r_i} \bmod N$, $R_i = (y_j^{F(ID_j)} + ID_j)^{r_i} \bmod N$, $f_i = F_1(m) \| F_2(F_1(m)) \oplus m$, $T_i = R_i \oplus f_i$, $t_i = H(m, e_i)$, and $s_i = r_i - x_i \cdot t_i \cdot f_i$. Afterward, U_i sends the signature $\sigma = \{e_i, s_i, T_i\}$ to U_j.

Message recovery phase: When U_j receives the signature, he firstly calculates $R_i' = e_i^{x_j}$ and $f_i' = R_i' \oplus T_i$. Then, U_j recovers the message m' by computing $m' = F_2([f_i']^{l_1}) \oplus [f_i']_{l_2}$, where $[f_i']^{l_1}$ denotes the left l_1 bits of f_i' and $[f_i']_{l_2}$ denotes the right l_2 bits of f_i'. Finally, U_j computes $t_i' = H(m', e_i)$ and checks the following equation: $e_i = g^{s_i} \cdot (y_i^{F(ID_i)} + ID_i)^{t_i' \cdot f_i'} \bmod N$. If this equation holds, U_j confirms and verifies the signature and signer's public key; otherwise, he rejects and terminates the procedure.

***Signature conversation phase*:** When U_j provides m', t_i', and e_i to a third-party as judge. The judge can verify the signature by computing $t_i' = H(m', e_i)$.

3.2 Demonstration of a New Type I Attack

Based on Wu and Xu's scheme, the demonstration of our attack is shown in Fig. 1 and can be formally described as follows.

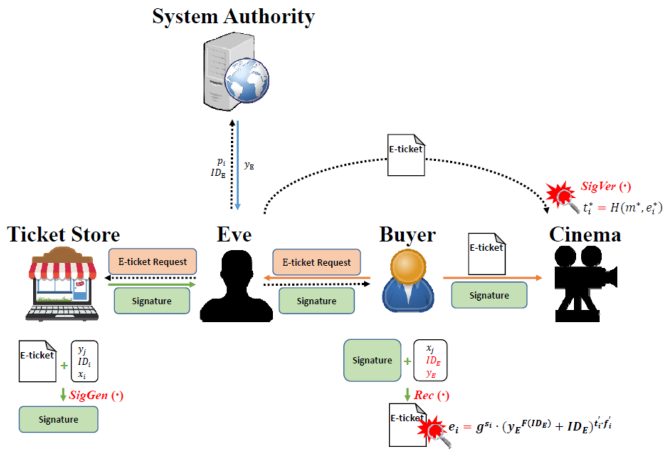


Fig. 1. The overview of an attack scenario.

An attack process: Let Eve be a probabilistic polynomial time (PPT) adversary. First, Eve aims at a particular signer, the *e*-ticket store, and obtains his public key $y_i = (p_i - ID)^{F^{-1}(ID_i)}$ mod N. By the definition of one-way hash function, Eve can calculate $y_i^{F(ID_i)} + ID_i \bmod N$ in polynomial time and, hence extract $p_i = g^{x_i} \bmod N$ from the *e*-ticket store's public key.

Next, Eve sends p_i and ID_E to SA, and receives his own public key $y_E = (p_i - ID_E)^{F^{-1}(ID_E)}$ mod N from SA, where ID_E is the identity of Eve.

Like phishing, Eve generates a fake internet store and waits for his victim Bob to request an *e*-ticket. By sending Bob's request, Eve receives a signature $\sigma = \{e_i, s_i, T_i\}$ from the *e*-ticket store. After that, Eve sends the signature σ to Bob. According to the procedure of the message recovery phase, Bob can successfully recover the message by computing $R'_i = e_i^{x_j}$ and $f_i = R'_i \oplus T_i$, and $m = F_2([f_i]^{(l_1)} \oplus [f_i]_{l_2}$. Then, Bob passes $e_i = g^{s_i} \cdot (y_E^{F(ID_E)} + ID_E)^{t'_i \cdot f'_i}$ mod N by using Eve's public key. In any *e*-commerce system, a customer will start to payment procedure immediately after she or he passes the verification of an *e*-ticket. Therefore, Eve is successful passes the signature verification and earns illegal income from a victim.

Moreover, we point out that a minor defect has existed in the signature conversation phase (also occurred in [11]). As mentioned by Wu and Xu, their scheme provides a signature conversation phase for negotiation when a debate occurs after the signature is verified in the message recovery phase. Unfortunately, equation $t_i' ? = H(m', e_i)$ only provides the integrity of m', t', and e_i but not verification of the signature. We use the same *e*-ticket example to explain why checking integrity is not enough to confirm the validity of a signature: Let Eve generate a random number e_i^* to compute $t_i^* = H(m_i^*, e_i^*)$. Here, m^* is an *e*-ticket. After receiving the tampered m^*, e_i^*, and t_i^*, the cinema can still pass the verification of $t_i^* = H(m^*, e_i^*)$. In this situation, Eve successfully passes the signature conversation phase. On the other hand, a signer can insist that an honest verifier publishes the fake message using the above-mentioned way to repudiate his signature when he denies his original message. Apparently, non-repudiation of the signature by their scheme is broken.

4. OUR PROPOSED SCHEME

Our proposed scheme has four phases: system initialization, signature generation, message recovery, and signature arbitration. At the beginning of our scheme, a system authority (SA) first runs the ***Setup*(.)** algorithm to initialize the system parameters and the ***KeyGen*(.)** algorithm for assisting users (signer and verifier) to create their public keys. When a signer wants to sign a message to a designated verifier, he/she runs the ***SigGen*(.)** algorithm to create the signature in the signature-generation phase. Next, a verifier can recover the message by the ***Rec*(.)** algorithm in the message recovery phase and verify the validity of the signature by the ***SigVer*(.)** algorithm. If it is necessary, the verifier can publish the message and the signature for arbitration. Hence, any third-party can verify it by using the algorithm in the last phase.

4.1 Algorithms of Our Proposed Scheme

There are five polynomial time algorithms (PTAs) executed by a system authority, a signer, a receiver, and verifiers in our proposed system. PTAs are started by a security parameter 1^k and work as follows:

***Setup*(.):** This algorithm takes 1^k, p, and q as inputs, where p and q are two big prime numbers. It returns a system parameter *Prarms*.

***KeyGen*(.):** This algorithm takes *Prarms*, a system authority's identity *SID*, a user's identity *ID*, and semi-public key p_{ID} as inputs. It returns the user public key y_{ID}.

***SigGen*(.):** This algorithm takes a message m, a signer's identity $S-ID$, a receiver's identity $R-ID$, a system authority's identity *SID*, a signer's private key $x_{S\text{-}ID}$, and a receiver's public key $y_{R\text{-}ID}$. It outputs a signature σ.

***Rec*(.):** This algorithm takes a signature σ, a signer's identity $S-ID$, a receiver's private key $x_{R\text{-}ID}$, and a signer's public key $y_{S\text{-}ID}$ as inputs. It outputs the original message m.

***SigVer*(.):** This deterministic algorithm takes a message m, a system authority's identity *SID*, a receiver's identity $R-ID$, a signer's public key $y_{S\text{-}ID}$, and a signature σ as inputs. It outputs a response (accept/reject) for the validity of this signature.

4.2 Four Phases of the Proposed Scheme

The details of the four phases are described as follows:

***System initialization phase*:** System authority (SA) takes two large primes p and q such that $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are also primes. Then, SA computes $N' = p' \cdot q'$ and chooses g, which is a base element of order N'. After that, SA keeps p, q, p' and q' secret and publishes N, g, and a one-way hash function $F()$: $\{0, 1\}^* \rightarrow \{0, 1\}^k$ to all users, where $N = p \cdot q$ and $k = \log(\min(p', q'))$. After SA finishes the initial setup, user U_i can choose a random number $x_i \in Z_N^*$ to compute $p_i = g^{x_i \cdot F(ID_i)} \bmod N$ and then submit a registration message $\{ID_i, p_i\}$ to SA. Subsequently, SA computes $y_i = (p_i)^{F^{-1}(SID)} \bmod N$ as U_i's public key and sends it back to U_i, where *SID* is the identity of SA. Finally, U_i can check the received public key by computing $y_i^{F(SID)} ? = p_i$. If they are equal, U_i accepts the public key; otherwise, U_i should re-run the procedure until the equation holds.

***Signature generation phase*:** When U_i wants to sign a message m to U_j, he/she chooses a random number $r_i \in_R Z_N^*$ to compute $e_i = g^{r_i} \bmod N$, $R_i = y_j^{F(SID) \cdot r_i} \bmod N$, $T_i = R_i \oplus m$, $t_i = H(m, e_i, ID_j)$, and $s_i = r_i - x_i \cdot F(ID_i) \cdot t_i$. After that, U_i sends the signature $\sigma = \{e_i, s_i, T_i\}$ to U_j.

***Message recovery phase*:** Once receiving σ from U_i, U_j computes $R_i' = e_i^{x_j \cdot F(ID_j)}$ and then retrieves the message by $m' = T_i \oplus R_i'$.

***Signature arbitration phase*:** U_j computes $t_i' = H(m', e_i, ID_j)$ and checks the equation: $e_i ? = g^{s_i} \cdot y_i^{F(SID) \cdot t_i'} \bmod N$. If the equation holds, U_j confirms the validity of signature and signer's public key; otherwise, he rejects and terminates the procedure. Moreover, if U_j publishes m', s_i, and e_i, anyone can verify the validity by computing $t_i' = H(m', e_i, ID_j)$ and $e_i ? = g^{s_i} \cdot y_i^{F(SID) \cdot t_i'} \bmod N$.

Theorem 1 (Correctness)**:** Our proposed scheme is correct.

***Proof*:** Assume that the original message m equals to the recovered message m' then

$$t_i = H(m, e_i, ID_j)$$
$$= H(m, e_i, ID_j) = t_i'$$

Correctness of the ***SigVer*(·)** is as follows:

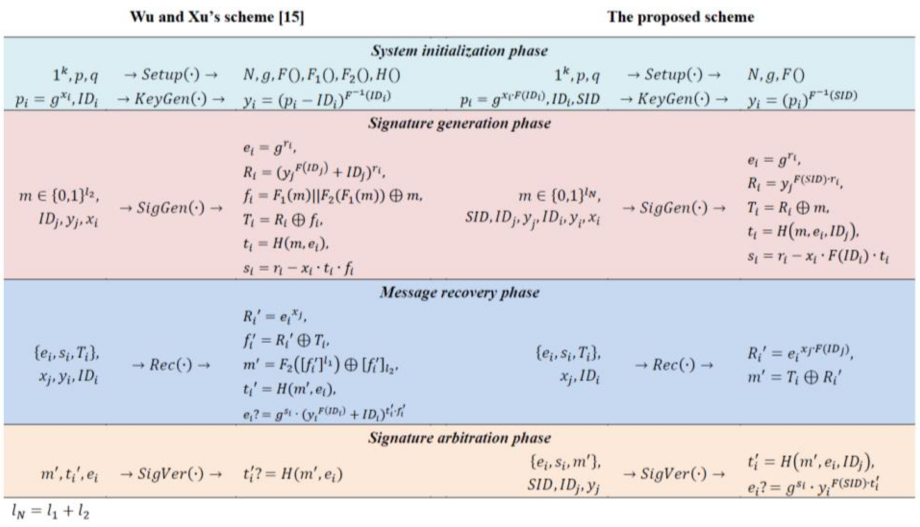
$$g^{s_i} \cdot y_i^{F(SID)\cdot t_i'} \bmod N$$
$$= g^{r_i - x_i \cdot F(ID_i) \cdot t_i} \cdot y_i^{F(SID)\cdot t_i'} \bmod N$$
$$= g^{r_i - x_i \cdot F(ID_i) \cdot t_i} \cdot p_i^{F^{-1}(SID)F(SID)\cdot t_i'} \bmod N$$
$$= g^{r_i - x_i \cdot F(ID_i) \cdot t_i} \cdot g^{x_i \cdot F(ID_i)\cdot t_i'} \bmod N$$
$$= g^{r_i - x_i \cdot F(ID_i) \cdot t_i + x_i \cdot F(ID_i)\cdot t_i'} \bmod N$$
$$= g^{r_i} \bmod N$$
$$= e_i.$$

Thus, e_i is equal to $g^{s_i} \cdot y_i^{F(SID)\cdot t_i'} \bmod N$ when $t_i = t_i'$. It means that verifier accepts $\sigma = \{e_i, s_i, T_i\}$ as a valid signature for the message m.

5. DIFFERENCE BETWEEN WU AND XU'S SCHEME AND OURS

In this section, we will elaborate on the difference between the scheme [10] and ours. Fig. 2 shows the comparisons of the two schemes, which include four main differences:

(1) In the system initialization phase of the proposed scheme, we put the user's identity into the parameter p_i for resisting the Type I attack. Details of the defense principle and the security proof are shown in Theorem 4 (Section 6).



Wu and Xu's scheme [15]			The proposed scheme		
		System initialization phase			
$1^k, p, q$	$\rightarrow Setup(\cdot) \rightarrow$	$N, g, F(), F_1(), F_2(), H()$	$1^k, p, q$	$\rightarrow Setup(\cdot) \rightarrow$	$N, g, F()$
$p_i = g^{x_i}, ID_i$	$\rightarrow KeyGen(\cdot) \rightarrow$	$y_i = (p_i - ID_i)^{F^{-1}(ID_i)}$	$p_i = g^{x_i \cdot F(ID_i)}, ID_i, SID$	$\rightarrow KeyGen(\cdot) \rightarrow$	$y_i = (p_i)^{F^{-1}(SID)}$
		Signature generation phase			
$m \in \{0,1\}^{l_2}$, ID_j, y_j, x_i	$\rightarrow SigGen(\cdot) \rightarrow$	$e_i = g^{r_i}$, $R_i = (y_j^{F(ID_j)} + ID_j)^{r_i}$, $f_i = F_1(m) \| F_2(F_1(m)) \oplus m$, $T_i = R_i \oplus f_i$, $t_i = H(m, e_i)$, $s_i = r_i - x_i \cdot t_i \cdot f_i$	$m \in \{0,1\}^{l_N}$, $SID, ID_j, y_j, ID_i, y_i, x_i$	$\rightarrow SigGen(\cdot) \rightarrow$	$e_i = g^{r_i}$, $R_i = y_j^{F(SID)\cdot r_i}$, $T_i = R_i \oplus m$, $t_i = H(m, e_i, ID_j)$, $s_i = r_i - x_i \cdot F(ID_i) \cdot t_i$
		Message recovery phase			
$\{e_i, s_i, T_i\}$, x_j, y_i, ID_i	$\rightarrow Rec(\cdot) \rightarrow$	$R_i' = e_i^{x_j}$, $f_i' = R_i' \oplus T_i$, $m' = F_2([f_i']^{l_1}) \oplus [f_i']_{l_2}$, $t_i' = H(m', e_i)$, $e_i ?= g^{s_i} \cdot (y_i^{F(ID_i)} + ID_i)^{t_i' \cdot f_i'}$	$\{e_i, s_i, T_i\}$, x_j, ID_i	$\rightarrow Rec(\cdot) \rightarrow$	$R_i' = e_i^{x_j \cdot F(ID_j)}$, $m' = T_i \oplus R_i'$
		Signature arbitration phase			
m', t_i', e_i	$\rightarrow SigVer(\cdot) \rightarrow$	$t_i' ?= H(m', e_i)$	$\{e_i, s_i, m'\}$, SID, ID_j, y_j	$\rightarrow SigVer(\cdot) \rightarrow$	$t_i' = H(m', e_i, ID_j)$, $e_i ?= g^{s_i} \cdot y_i^{F(SID)\cdot t_i'}$

$l_N = l_1 + l_2$

Fig. 2. Flow structure of Wu and Xu's scheme and the proposed scheme.

(2) In the signature generation phase of the proposed scheme, we put the receiver's identity into the parameter t_i for resisting the Type II attack (proposed by [11]). Details of the

defense principle and the security proof are shown in Theorem 5 (Section 6).

(3) In the signature generation phase of the proposed scheme, we modify the method of message embedding for making the length of the signing message longer. Details of comparison of message length are shown in Table 1 (Section 7).

(4) In Wu and Xu's scheme, only the specific receiver can confirm the validity of a signature in the message recovery phase. However, the signature cannot be verified by a third-party in the signature conversation phase due to the reason described in Subsection 3.2. Hence, we redefined the goals of the message recovery phase and the signature conversation phase for making the signature scheme more robust. In the message recovery phase of the proposed scheme, the receiver recovers the message from a signature via the ***Rec*(.)** algorithm. When the receiver publishes the message and the signature, anyone including him/herself can confirm the validity of the signature by using the ***SigVer*(.)** algorithm now.

6. SECURITY MODELS AND PROOFS

In this section, we will prove our system is secure under the random oracle model (ROM) [12] based on two security notions: 1) the confidentiality of the self-certified signature with message recovery under adaptive chosen-message attacks (CON-SCMRa CMA); 2) the unforgeability of the self-certified signature with message recovery under adaptive chosen-message attacks (UNF-SCMR-aCMA). Before we show that our proposed scheme is provably secure on ROM, the discrete logarithm problem, one-way hash function, and forking lemma [13] are introduced first. Then, we will prove that our proposed scheme is against Type I and Type II attacks, as mentioned in Subsection 2.1.

Definition 1 (*Discrete logarithm problem* (*DLP*))**:** Let $^*_p$ be a finite multiplicative cycle group, p a prime number, and g a generator of $^*_p$. Then, the discrete logarithm problem [14] can be defined as follows: Given a group $^*_p$, a generator g, an integer number b. To find a number a in $^*_p$ such that $b = g^a \bmod p$ is a very hard problem when the prime p is large.

Definition 2 (*One-way Hash Function*)**:** A function $h : X \rightarrow Y$ is called a one-way hash function [15] if and only if it satisfies the following five properties:

***Compression*:** Given an arbitrary length binary string $X \in \{0, 1\}^*$ as input, h outputs a fixed length binary string $Y \in \{0, 1\}^n$ such that $Y = h(X)$, where Y is the hash digest of input X.

***Ease of computation*:** Given an arbitrary length binary string $X \in \{0, 1\}^*$ and a one-way hash function h such that $Y = h(X)$, h can be computed in polynomial time.

***Preimage resistance*:** Given a fixed length binary string $Y \in \{0, 1\}^n$ and a one-way hash function h, there exists no polynomial algorithm to find $X \in \{0, 1\}^*$ that satisfies $Y = h(X)$.

***2nd-preimage resistance*:** Given an arbitrary length binary string $X \in \{0, 1\}^*$ and a oneway hash function h, there exists no polynomial algorithm to find $X' \in \{0, 1\}^*$ such that $h(X) = h(X')$, where $X \neq X'$.

***Collision resistance*:** Finding two different inputs binary strings X, $X' \in \{0, 1\}^*$ such that $h(X) = h(X')$ cannot be successful computed in polynomial time.

Definition 3 (Forking lemma)**:** Assume that a probabilistic polynomial Turing machine A can produce a valid signature $\{\sigma_1, h, \sigma_2\}$ on message m with security parameter k, within a time bound T, and with probability $\varepsilon \geq 10(q_s + 1)(q_s + q_h)/2^k$, where q_s and q_h denote the number of signing queries and the number of random oracle queries. If the signature can be simulated without knowing the private key, within an indistinguishable distribution probability, then there exists another Turing machine A', can produce two valid signatures $\{\sigma_1, h, \sigma_2\}$ and $\{\sigma', h', \sigma_2'\}$ on the same message m such that $h \neq h'$ in expected time $T' \leq 120686 \cdot q_h \cdot T/\varepsilon$.

Definition 4 (The CON-SCMR-aCMA Game)**:** Let A be a probabilistic polynomial time adversary. A interacts with a challenger C in the following game.

Initialization. A chooses two identities $\{ID_0, ID_1\}$ and two messages $\{m_0, m_1\}$, where the lengths of m_0 and m_1 are the same. A sends $\{ID_0, ID_1\}$ to C.

Queries. A issues the following queries in any adaptive way:
- Setup queries. This oracle takes 1^k, p, and q as inputs, where p and q are two big prime numbers. It returns the system parameter *Prarms*.
- Extract queries. This oracle takes *Prarms* and a user identity $ID_i \notin \{ID_0, ID_1\}$ as inputs. It returns the corresponding private key sk_i. After receiving sk_i, C sends $\{ID, sk_i\}$ to A.
- Sign queries. This oracle takes a message $m_i \notin \{m_0, m_1\}$ and a target public key pk_i as inputs. It returns a signature σ_i. After receiving σ_i, C sends σ_i to A.

Challenge. A sends messages $\{m_0, m_1\}$ to C. C flips a fair coin $\lambda \in \{0, 1\}$. If $\lambda = 0$, C generates two signatures $\{\sigma_0, \sigma_1\}$ on messages $\{m_\lambda, m_{1-\lambda}\}$ for receivers $\{ID_\lambda, ID_{1-\lambda}\}$; otherwise, C generates two signatures $\{\sigma_0, \sigma_1\}$ on messages $\{m_\lambda, m_{1-\lambda}\}$ for receivers $\{ID_{1-\lambda}, ID_\lambda\}$. After that, C sends $\{\sigma_0, \sigma_1\}$ to A.

Guess. A outputs one bit $\lambda' \in \{0, 1\}$. If $\lambda' = \lambda$, it means that A has linked messages and identities by signatures and wins the CON-SCMR-aCMA Game. The success probability of a scheme is defined as $Succ^{link} = [2Pr[\lambda' = \lambda] - 1]$. The CON-SCMR-aCMA is confidential if for all probabilistic PTA A, A wins the above game with negligible probability.

Theorem 2 (Confidentiality)**:** Our proposed scheme is CON-SCMR-aCMA secure under ROM if DLP assumption holds.

***Proof*:** By Definition 4, we assume that in probabilistic polynomial time, adversary A obtains two signatures $\{\sigma_0, \sigma_1\}$ from the challenger C after playing the game. To prove the confidentiality, we first show that $\{e_i, s_i, T_i\} \in \{\sigma_0, \sigma_1\}$ as follows:

$$e_i = g^{r_i} \bmod N \tag{3}$$
$$s_i = r_i - x_i \cdot F(ID_i) \cdot t_i \tag{4}$$
$$T_i = (p_i)^{F^{-1}(SID) \cdot F(SID) \cdot r_i} \oplus m \tag{5}$$

We can derive the following equations from Eqs. (2)-(4):

$$r_i = \log_g e_i \bmod N \tag{6}$$

$$x_i = (s_i - \log_g e_i)/(F(ID_i) \cdot H(m_i, e_i, ID_j)) \tag{7}$$

$$m_i = T_i \oplus g^{x_i \cdot r_i} \tag{8}$$

Because ID_0, ID_1, m_0, and m_1 are picked by A, A can easily compute $F(ID_0)$, $F(ID_1)$, $H(m_0, e_i, ID_0)$, $H(m_0, e_i, ID_1)$, $H(m_1, e_i, ID_0)$, and $H(m_1, e_i, ID_1)$. However, A cannot extract r_i, x_i, and m_i from Eqs. (5)-(7) under DLP assumption. In addition, A never performs the extract queries for $\{ID_0, ID_1\}$.

Finally, A outputs one bit $\lambda' \in \{0, 1\}$. If $\lambda' = \lambda$, A has linked messages and identities by signatures and wins. In our scheme, A guesses the probability of $Pr[\lambda' = 0]$ and $Pr[\lambda' = 1]$ which is uniform, *i.e.*, $Pr[\lambda' = \lambda] = 1/2$ and $Succ^{link}_{CON-SCMR-aCMA} = 0$. Hence, A links neither signatures with the corresponding messages nor the messages with the corresponding identities. Thus, the success probability is negligible and our scheme is CON-SCMR-aCMA secure.

Theorem 3 (Unforgeability)**:** If an adversary A can forge a signature in our proposed SCMR scheme within probabilistic polynomial time T and with non-negligible probability $\varepsilon \geq 10(q_s + 1)(q_s + q_h)/2^k$, where $k = \log(\min(p', q'))$, then a challenge C, can solve the DLP within probabilistic polynomial time T' and with non-negligible probability ε' such that $\varepsilon' \geq 1/9$, $T' \leq 120686 \cdot q_h \cdot T/\varepsilon + (2q_s + 3)t_e + (q_s + 2)t_m$. Here, q_s, q_h, t_e, and t_m denote the number of signing queries, the number of random oracle queries, the time of a modular exponentiation computation, and the time of a multiplication computation, respectively.

Proof**:** Let A be a probabilistic polynomial time adversary. A interacts with a challenger C in the following UNF-SCMR-aCMA game.

Initialization: Here, we assume that C already finishes the setup-queries and assigns ID_0 and ID_1 as the signer and the receiver, respectively, to simplify the UNF-SCMRaCMA game. After setting up, C sends $\{ID_0, p_0\}$ and $\{ID_1, p_1, x_1\}$ to A, where parameters $p_0, p_1 \in Z_n^*$ and x_1 is the private key of ID_1. Now, C is ready to answer the queries.

Queries. A issues the following queries in any adaptive way:

- H-queries. This oracle takes any hash function as inputs, then it returns the corresponding hash value. When receiving a request (at most q_h times), C firstly checks the h-list by the corresponding identity. If the hash value h_i exists, C returns it to A; otherwise, C chooses a number $r_i \in_R Z_n^*$ as a response and stores $\{ID_i, r_i\}$ into h-list.
- Sign-queries. In this oracle, A sends m_i to C at most q_s time. C first looks up the s-list. If the corresponding signature $\sigma_i = \{e_i, s_i, T_i\}$ exists, C then returns σ_i to A; otherwise, C randomly chooses a number r_i and computes $e_i = g^{r_i} \bmod N$, $R_i = y_j^{F(SID) \cdot r_i} \bmod N$, $T_i = R_i \oplus m_i$, $t_i = H(m_i, e_i, ID_j)$, and $s_i = r_i - x_i \cdot F(ID_i) \cdot t_i$. After that, C sends the signature $\sigma_i = \{e_i, s_i, T_i\}$ to A and stores $\{m_i, e_i, s_i, T_i\}$ into s-list.

Results. A successfully generates a valid signature $\sigma_0 = \{e_0, s_0, T_0\}$ on an unsigned message m_0. If m_0 and ID_0 never appear in the s-list and the h-list, C then computes $R'_0 = e_0^{x_0 F(ID_1)}$

and retrieves the message by $m'_0 = T_0 \oplus R'_0$. Additionally, C computes $t'_0 = H(m'_0, e_0, ID_1)$ and checks the following equation: $e_0 \; ? = g^{s_0} \cdot y_0^{F(SID) \cdot t'_0}$ mod N. If they are equal, C accepts the signature; otherwise, C terminates the oracle procedure.

According to Definition 3, C can generate another signature $\sigma_1 = \{e_1, s_1, T_1\}$ on the same message m_0 with the same random tape ω. Therefore, C obtains two equations.

$$s_0 = r_0 - x_0 \cdot F(ID_0) \cdot t_0 \bmod N \tag{9}$$

$$s_1 = r_0 - x_0 \cdot F(ID_0) \cdot t_1 \bmod N \tag{10}$$

By calculating Eq. (8) minus Eq. (9), C obtains $x_0 = (s_0 - s_1)/F(ID_0) \cdot (t_1 - t_0)$, which is a solution to DLP with non-negligible probability $\varepsilon' \geq 1/9$. The total running time of solving DLP is $120686 \cdot q_h \cdot T/\varepsilon$ with t_e and t_m in replying every sign-queries and the verification of C. We summarize that C spends $2t_e + t_m$ in replying every sign-queries and $3t_e + 2t_m$ in the verification. Hence, C solves DLP with $T' \leq 120686 \cdot q_h \cdot T/\varepsilon + (2q_s + 3)t_e + (q_s + 2)t_m$. Here, q_s, q_h, t_e and the probability $\varepsilon' \geq 1/9$.

Theorem 4 (Anti-Type I attack)**:** Our proposed scheme is against Type I attack.

***Proof*:** Let A be a probabilistic polynomial time (PPT) adversary. First, A aims at a particular signer U_i and obtains his public key $y_i = (p_i)^{F^{-1}(SID)}$ mod N. Then, A can calculate $y_i^{F(SID)}$ mod N in polynomial time and extract $p_i = g^{x_i \cdot F(ID_i)}$ mod N.

Observation 1: The parameter p_i includes $F(ID_i)$, and A cannot remove it from parameter $p_i = g^{x_i \cdot F(ID_i)}$ mod N without SA's secure parameters p and q.

Hence, A only has two options to continue his attack: (1) Replays original parameter $p_i = g^{x_i \cdot F(ID_i)}$ mod N to SA; (2) Computes $p'_A = p_i^{F(ID_A)} = g^{x_i \cdot F(ID_i) \cdot F(ID_A)}$ mod N and sends to SA for registration.

Observation 2: Both public keys $y_A = y_i = (p_i)^{F^{-1}(SID)}$ mod N and $y'_A = (p'_A)^{F^{-1}(SID)}$ mod N still bind with $F(ID_i)$. For example, the parameter $F(ID_i)$ is hidden inside parameters p_i and p'_A.

Now, A executes the same phishing attack as mentioned in Section 3.2. This time, in our proposed scheme, a receiver U_j can easily detect the e-ticket is unauthorized by checking the equation $g^{s_i} \cdot y_A^{F(SID) \cdot t'_i} = e_i$.

Observation 3: In this equation, we can see that $s_i = r_i - x_i \cdot F(ID_i) \cdot t_i$ also includes the parameter $F(ID_i)$. Additionally, A cannot generate $s_A = r_i - x_i \cdot F(ID_A) \cdot t_i$ by himself because he has no x_i's information.

Based on three observations, a receiver U_j can easily detect the e-ticket is unauthorized and terminates the transaction protocol when U_j finds out the equation $g^{s_i} \cdot y_A^{F(SID) \cdot t'_i}$ is not equals to e_i. Hence, our proposed scheme resists the Type I attack.

Theorem 5 (Anti-Type II attack)**:** Our proposed scheme is against Type II attack.

***Proof*:** Let A and C be two probabilistic polynomial time adversaries, and C shares his/her private key x_C with A. According to the attack scenario of [11], A first obtains a valid signature $\sigma = \{e_i, s_i, T_i\}$ from a signer U_i in our proposed scheme. Then, A computes $R_i^* = g^{r_i \cdot x_C} \bmod N$ and $T_i^* = R_i^* \oplus m$. After that, A sends the forge signature $\sigma^* = \{e_i, s_i, T_i^*\}$ to C. However, this attack will not success when encountering our proposed scheme due to the following reasons. When C publishes $\{m, e_i, s_i\}$, the verifier first calculates $t_i^* = H(m, e_i, ID_C)$. However, t^* is not equal to the original $t_i' = H(m, e_i, ID_A)$. Hence, this forge signature cannot pass the equation check as $g^{s_i} \cdot y_i^{F(SID) \cdot t_i'} \bmod N ? = g^{s_i} \cdot y_i^{F(SID) \cdot t_i^*} \bmod N$ when $t_i' \neq t_i^*$. As a result, our proposed scheme resists the Type II attack.

7. PERFORMANCE ANALYSIS

In this section, we present the communication and computation costs of our proposed scheme and the comparisons to related work. Additionally, we evaluate the performances of our proposed scheme in terms of run-time in a complete *e*-ticket system (Fig. 1).

Before we compute the communication cost, the length of our signature needs to be analyzed. First, we follow the suggestion by the RSA Laboratories in 2016, where the security requirement of an RSA key length should be 3072 bits beyond 2030 [16]. Therefore, we set the group size (N) as 3072 bits. Now, our signature consists of three parameters, which are $e_i = g^{r_i} \bmod N$, $s_i = r_i - x_i \cdot F(ID_i) \cdot t_i$, and $T_i = R_i \oplus m$. Apparently, the total length is $3N$, which equals to 9216 bits. In an environment of 1Mb/s upload/download speed, it takes around 90 millisecond (*ms*) for sending/receiving a signature.

The computation cost of our proposed scheme can be divided into four parts. The first part is the system initialization phase (S-I). It takes $3t_e + 3t_m + 3t_h + 1t_{mi}$, where t_e, t_m, t_h, and t_{mi} are the operation times of one modular exponentiation with modulo N, one scalar multiplication with modulo N, a one-way hash function, and one modular inversion with modulo, respectively. The second part is the signature generation phase (S-G). It takes $2t_e + 3t_m + 3t_h$. Here, we omit the computation time of the exclusive-OR operation which is negligible. Finally, $t_e + t_m + t_h$ and $2t_e + 1t_m + 2t_h$ are the computation times taken in the message recovery phase (M-R) and the signature arbitration phase (S-A), respectively.

Table 1 shows the comparisons of signature length and message length between ours and [2, 4, 5, 7, 10, 11]. Despite the signature's length of Tseng *et al.* [2] and Lv *et al.* [7] is shorter, our scheme provides superior security compared to theirs (Remarks: the scheme of [2, 7] are insecure as mentioned in Section 2). Additionally, it is worth mentioning that the signature length of schemes proposed by Wu and Xu [10] and Sadeghpour [11] are the same as ours but our scheme supports longer input message. More precisely, the input message length is $|Z_N|$ in our scheme, but that in [10, 11] are $|F_2|$, where
$F_2 : \{0, 1\}^* \rightarrow \{0, 1\}^{l_2}$ and $|l_2| < |Z_N|$.

Table 1. Comparisons of signature/message length.

Scheme	Signature Length (group elements)	Message Length
Ours	$3\|Z_N\| = 9216$*bits*	$\|Z_N\| = 3072$*bits*
[11]	$3\|Z_N\| = 9216$*bits*	$\|F_2\| < 3072$*bits*
[10]	$3\|Z_N\| = 9216$*bits*	$\|F_2\| < 3072$*bits*
[5]	$3\|Z_N\| = 9216$*bits*	$\|Z_N\| = 3072$*bits*
[7]	$2\|Z_N\| + \|H\|^* = 6400$*bits*	$\|Z_N\| = 3072$*bits*
[4]	$3\|Z_N\| = 9216$*bits*	$\|Z_N\| = 3072$*bits*
[2]	$2\|Z_N\| = 6144$*bits*	$\|Z_N\| = 3072$*bits*

* We assume that $|H|$ is 256 bits (SHA-256)

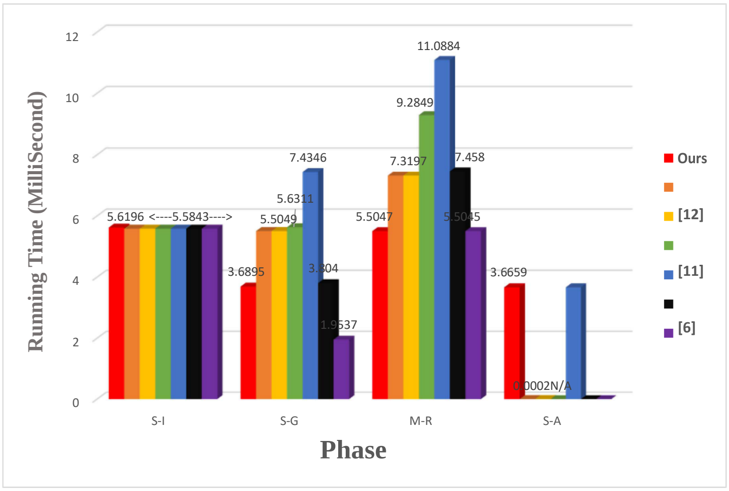


Fig. 3. Evaluation of execution time of our scheme and the related works. (N/A means that the scheme does not support the signature arbitration/conversation phase).

Table 2. Comparisons of computation cost.

Scheme	S-I	S-G	M-R	S-A
Ours	$3t_e + 3t_m + 3t_h + t_{mi}$	$2t_e + 3t_m + 3t_h$	$3t_e + 2t_m + 3t_h$	$2t_e + 1t_m + 2t_h$
[11]	$3t_e + 2t_h + t_{mi}$	$3t_e + 2t_m + 4t_h$	$4t_e + t_m + 2t_h$	t_h
[10]	$3t_e + 2t_h + t_{mi}$	$3t_e + 2t_m + 4t_h$	$4t_e + t_m + 2t_h$	t_h
[5]	$3t_e + 2t_h + t_{mi}$	$3t_e + 4t_m + 2t_h + t_{mi}$	$5t_e + 4t_m + 2t_h + t_{mi}$	*N/A*
[7]	$3t_e + 2t_h + t_{mi}$	$4t_e + 2t_m + 2t_h + t_{mi}$	$6t_e + 2t_m + 2t_h + t_{mi}$	$2t_e + t_m + t_h$
[4]	$3t_e + 2t_h + t_{mi}$	$2t_e + 4t_m + t_h + t_{mi}$	$4t_e + 4t_m + 2t_h + t_{mi}$	*N/A*
[2]	$3t_e + 2t_h + t_{mi}$	$t_e + 2t_m + t_h + t_{mi}$	$3t_e + 2t_m + 3t_h$	$2t_e + 1t_m + 2t_h$

N/A means that the scheme does not support the S-A phase

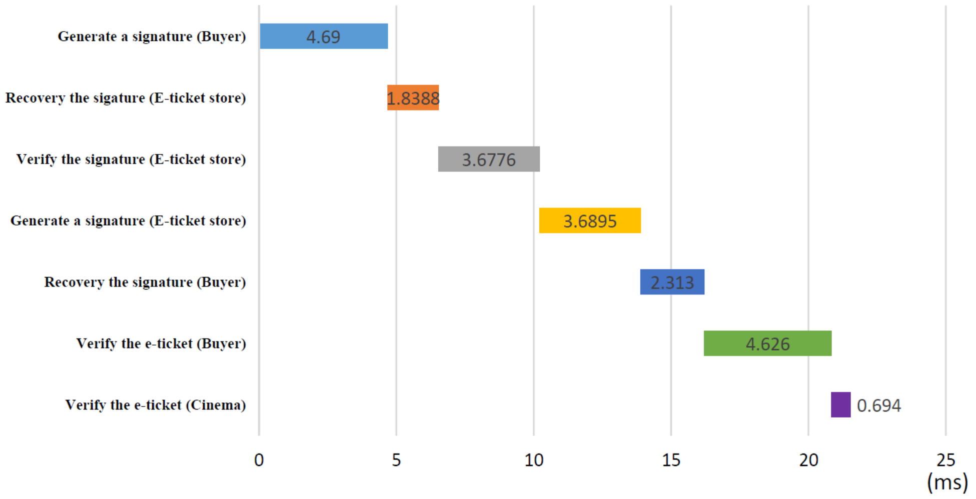


Fig. 4. Gantt chart of a complete *e*-ticket system.

Table 3. Running times of related operations.

Operations	t_e	t_m	t_h
E-ticket store[1]	1.8269	0.0117	0.0002
Cinema[2]	0.339	0.001	0.007
Buyer[3]	2.249	0.008	0.056

[1] MIRACL library on Inter(R) Core(TM)2 T6570 2.1GHz with 4G memory and the Windows7 32bit operating system [10].

[2] MIRACL library on Inter(R) Core(TM) i5-4460S 2.90GHz with 4G memory and Windows 8 operating system [17].

[3] MIRACL library on Samsung Galaxy S5 with a Quad-core 2.45G processor, 2G memory and the Android 4.4.2 operating system [17].

Table 2 shows the comparisons of computation costs [2, 4, 5, 7, 10, 11] and ours. Here, our cost of M-R is $3t_e + 2t_m + 3t_h$ instead of the original cost $t_e + t_m + t_h$. The reason is that, the results in literature [2, 4, 5, 7, 10, 11] gave receiver immediate verification of the signature in their message recovery phases. On the other hand, our proposed scheme only recovers the message. To have a fair comparison, we add the cost of S-A to that of M-R.

In addition, the time of t_e, t_m, t_h, and t_{mi} [10] is about 1.8269*ms*, 0.0117*ms*, 0.0002*ms*, and 0.1032*ms*, respectively. Hence, our scheme takes 5.6196*ms* in the system initialization phase, 3.6895*ms* in the signature generation phase, 5.5047*ms* in the message recovery phase (Remarks: this phase overlaps the time of the signature arbitration phase), and 3.6659*ms* in the signature arbitration phase. Fig. 2 presents the execution time in S-I phase of the proposed scheme is similar to all schemes, while the execution time in S-G and M-R phases of ours is faster than [4, 5, 7, 10, 11]. Moreover, only our scheme and the scheme [7] provide a completely S-A phase.

In our experiments, we evaluate the performances of our proposed scheme in terms of run-time in a complete *e*-ticket system, which mentioned in Section 1 (Fig. 1). We summarize the time in milliseconds by the simulation used in [10, 17], and perceive (1) Inter(R) Core(TM)2 T6570 2.1GHz with 4G memory and the Windows7 32bit operating system as an *e*-ticket store; (2) Inter(R) Core(TM) i5-4460S 2.90GHz with 4G memory and Windows 8 operating system as a cinema; and (3) Samsung Galaxy S5 with a Quadcore 2.45GHz processor, 2G memory and the Android 4.4.2 operating system as a buyer. The simulation results of the average running time are listed in Table 3.

Here, we omitted the run-time of the system initialization phase due to it is part of the preparation and it only needs to perform once. In fact, users spend most of their executing time on running other phases in practice. Hence, the procedure will start with a buyer who wants going to a cinema and generates a signature on his/her request message. It takes the buyer $2t_e + 3t_m + 3t_h = 4.69ms$ to generate a signature. After that, it takes the *e*-ticket store $1t_e + 1t_m + 1t_h = 1.8388ms$ and $2t_e + 2t_m + 2t_h = 3.6776ms$ to recover the request message and verify the signature, respectively. Once the signature with the request message is verified, it takes the *e*-ticket store $2t_e + 3t_m + 3t_h = 3.6895ms$ to generate a second signature on an *e*-ticket. Then, it takes the buyer $1t_e + 1t_m + 1t_h = 2.313ms$ to recover the *e*-ticket and $2t_e + 2t_m + 2t_h = 4.626ms$ to verify the ticket and the signature. When the buyer requests a movie service, it takes the cinema $2t_e + 2t_m + 2t_h = 0.694ms$ to verify an *e*-ticket. Totally, it only costs 21.5289*ms* from a buyer request an *e*-ticket to he/she watch a movie in a cinema. The Gantt chart of the run-time of an *e*-ticket system as shown in Fig. 3. Therefore, the results show that our scheme has good performance in terms of run-time and is suitable for the real-time applications.

8. CONCLUSIONS

In this paper, we have presented a new security-threat situation that could crash the security of the existing countermeasures on the self-certified digital signature scheme with message recovery. We have proposed a new scheme to rectify this problem. Our scheme achieves a high security level with low computation and communication costs; further, this scheme is suitable for practical use in the *e*-commerce systems, *e.g.*, *e*-tickets, pay-tv, *e*-bidding. In the future, we will upgrade our signature scheme to support the techniques of the batch and group verification.

REFERENCES

1. M. Girault, "Self-certified public keys," in *Proceedings of Workshop on the Theory and Application of Cryptographic Techniques*, 1991, pp. 490-497.
2. Y. Tseng, J. Jan, and H. Chien, "Digital signature with message recovery using self-certified public keys and its variants," *Applied Mathematics and Computation*, Vol. 136, 2003, pp. 203-214.
3. Z. Shao, "Improvement of digital signature with message recovery using self-certified public keys and its variants," *Applied Mathematics and Computation*, Vol. 159, 2004, pp. 391-399.
4. Y. Chang, C. Chang, and H. Huang, "Digital signature with message recovery using self-certified public keys without trustworthy system authority," *Applied Mathematics and Computation*, Vol. 161, 2005, pp. 211-227.
5. E. Yoon and K. Yoo, "An improved digital signature with message recovery using self-certified public keys without trustworthy system authority," in *Proceedings of the 32nd Conference on Current Trends in Theory and Practice of Computer Science*, 2006, pp. 548-555.
6. J. Zhang, W. Zou, D. Chen, and Y. Wang, "On the security of a digital signature with message recovery using self-certified public key," *Informatica (Slovenia)*, Vol. 29, 2005, pp. 243-346.
7. J. Lv, X. Wang, and K. Kim, "Practical convertible authenticated encryption schemes using self-certified public keys," *Applied Mathematics and Computation*, Vol. 169, 2005, pp. 1285-1297.
8. Z. Shao, "Cryptanalysis and improvement of practical convertible authenticated encryption schemes using self-certified public keys," *Informatica*, Vol. 17, 2006, pp. 577-586.
9. J. Zhang, H. Chen, S. Gao, and Q. Geng, "Comment on a digital signature scheme with using self-certified public keys," in *Proceedings of International Forum on Information Technology and Applications*, 2009, pp. 678-680.
10. F. Wu and L. Xu, "An improved and provable self-certified digital signature scheme with message recovery," *International Journal of Communication Systems*, Vol. 28, 2015, pp. 344-357.

11. M. Sadeghpour, "Cryptanalysis and modification of an improved selfcertified digital signature scheme with message recovery," *American Scientific Research Journal for Engineering, Technology, and Sciences*, Vol. 28, 2017, pp. 257-265.
12. M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *Proceedings of the 1st ACM Conference on Computer and Communications Security*, 1993, pp. 62-73.
13. D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *Journal of Cryptology*, Vol. 13, 2000, pp. 361-396.
14. D. Stinson, *Cryptography: Theory and Practice*, 3rd ed., ser. Discrete Mathematics and Its Applications, Taylor & Francis, 2005.
15. A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, ser. Discrete Mathematics and Its Applications, Taylor & Francis, 1997.
16. B. Kaliski, "Rsa laboratories: Twirl and rsa key size," 2003, https://www.semanticscholar.org/paper/TWIRL-and-RSA-KeySize-Kaliski/cb6cb50d6330f58606f8f14ed2d8f6f96f221075.
17. L. Wu, J. Wang, K. R. Choo, and D. He, "Secure key agreement and key protection for mobile device user authentication," *IEEE Transactions on Information Forensics and Security*, Vol. 14, 2019, pp. 319-330.

Chin-Yu Sun (孫勤昱) received the MS degree in Department of Information Engineering and Computer Science from Feng Chia University, Taichung, Taiwan in 2013. He is currently pursuing his Ph.D. degree in Computer Science from National Tsing Hua University, Hsinchu, Taiwan. His research interests include cryptography, quantum cryptography, information security, and security in electronic commerce, mobile communication, and cloud computing.

Hsiao-Ling Wu (吳曉玲) received her Ph.D. degree in Information Engineering and Computer Science from Feng Chia University, Taichung, Taiwan in 2018. She is also the honorary member of The Phi Tau Phi Scholastic Honor Society Taiwan in the same year. She received the BS degree in Department of Applied Mathematics from Feng Chia University, Taichung, Taiwan in 2007. Now she is as a Postdoctoral Fellow in Department of Information Management from Chaoyang University of Technology, Taichung, Taiwan. Her current research interests include electronic commerce, information security, image processing, cryptography, and mobile communications.

Hung-Min Sun (孫宏民) received his Ph.D. degree in Computer Science and Information Engineering from National Chiao Tung University in 1995. Currently, he is working as the Director of Institute of Information Systems and Applications, and a Full Professor with the Department of Computer Science, National Tsing Hua University.

TingTing Hwang (黃婷婷) received the M.S. and Ph.D. degrees in Computer Science from Pennsylvania State University, University Park, PA, USA, in 1986 and 1990, respectively. She is currently a Professor with the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan. Her current research interests include logic synthesis/optimization and high-level synthesis.