An efficient certificateless short signature scheme from pairings

Debiao He*
School of Mathematics and Statistics, Wuhan University, Wuhan, 430072 China,

Abstract*:* To avoid the inherent key escrow problem in ID-based public key cryptosystem, Al-Riyami and Paterson introduced a new approach called certificateless public key cryptography. Recently, several short certificateless signature schemes are presented to improve the performance. In this paper, we propose an efficient short certificateless signature scheme which is secure against the super adversary. Compared with the related scheme, our scheme has the best performance in both sign algorithm and the verify algorithm.

Key words: Certificateless public key cryptography; Shrot signature; Bilinear pairings; Provable security

1. Introduction

Public-key cryptography(PKC) has become one of the essential techniques in providing security services in modern communications. In traditional public-key cryptosystems, a pair of public/private keys should be computed by each user. Since the public key is a string of random bits, a digital certificate of the public key is required to provide public-key authentication. Anyone who wants to send messages to others must obtain their authorized certificates that contain the public key. However, this requirement brings lots of certificate management problems in practice.

In order to simplify the public-key authentication, Shamir [1] introduced the concept of identity-based (ID-based) cryptosystem problem. In this system, each user needs to register at a key generator centre (KGC) with identify of himself before joining the network. Once a user is accepted, the KGC will generate a private key for the user and the user's identity (e.g. user's name or email address) becomes the corresponding public key. In this way, in order to verify a digital signature or send an encrypted message, a user only needs to know the "identity" of his communication partner and the public key of the KGC. However, this cryptosystem involves a KGC, which is responsible for generating a user's private key based on his identity. As a result, the KGC can literally decrypt any ciphertext or forge any user's signature on any message. To avoid the inherent key escrow problem in ID-based public key cryptosystem, Al-Riyami and Paterson [2] introduced a new approach called certificateless public key cryptography (CLPKC). The CLPKC is intermediate between traditional PKC and ID-based cryptosystem. In a certificateless cryptosystem, a user's private key is not generated by the KGC alone. Instead, it consists of partial private key generated by the KGC and some secret value chosen by the user. So, the KGC is unable to obtain the user's private key. In such a way that the key escrow problem can be solved. Intuitionally, CLPKC has nice features borrowed from both ID-based cryptography and traditional PKC. It alleviates the key escrow problem in ID-based cryptography and at the same time reduces the cost and simplifies the use of the technology when compared with traditional PKC.

Following the pioneering work due to Al-Riyami and Paterson [2], several certificateless signature (CLS) schemes [3-8] have been proposed. However, certificateless signatures generated

*Corresponding author.
E-mail: hedebiao@163.com, *Tel*:+008601530718492

by schemes [2-7] have approximately 320-bit sizes and signatures in [8] have at least 480-bit sizes if using an elliptic curve on F_3^{97}. Because of the small size of short signatures, they are needed in environments with stringent bandwidth constraints, such as bar-coded digital signatures on postage stamps. Hence, it's necessary for us to construct a short CLS scheme.

In 2009, Du et al.[9] presented the first short CLS scheme that is proved to be secure in the random oracle model under the hardness assumption of the collusion attack algorithm with k traitor (k-CAA) [10] and the inverse computational Diffie-Hellman(Inv-CDH) problem. Recently, Choi et al. [11] demonstrated Du et al.'s scheme is insecure against the Type 1 adversary, which can carry out the replace public key queries. They also proposed a CLS scheme and prove that their scheme is provably secure in the random oracle model under the computational Diffie-Hellman (CDH) assumption.

In this paper, we present an efficient certificateless short signature scheme inspired by Zhang et al.'s work[12]. We also prove that our scheme is provably secure under the random oracle model. Compared with the related scheme, our scheme is most efficient. Then our scheme is more suitable for the practical applications

The rest of the paper is organized as follows: Section 2 introduces some preliminaries used in this paper, Section 3 proposes our scheme and discusses the security analysis of our scheme, Section 4 provides performance features of the presented scheme, and at the end, Section 5 concludes this paper.

2. Preliminaries

2.1.Mathematical background

Let G_1 be a cyclic additive group of prime order q, and G_2 be a cyclic multiplicative group of the same order q. We let P denote the generator of G_1. A bilinear pairing is a map $e: G_1 \times G_1 \to G_2$ which satisfies the following properties:

(1) Bilinearity

$e(aQ, bR) = e(Q, R)^{ab}$, where $Q, R \in G_1$, $a, b \in Z_q^*$.

(2) Non-degeneracy

$e(P, P) \neq 1_{G_2}$.

(3) Computability

There is an efficient algorithm to compute $e(Q, R)$ for all $Q, R \in G_1$.

The Weil and Tate pairings associated with supersingular elliptic curves or abelian varieties can be modified to create such admissible pairings, as in [9]. The following problems are assumed to be intractable within polynomial time.

Definition 1. k-CCA[10]. For an integer k, and $s \in Z_q^*$, $P \in G_1$. Given

$\{P, sP, e_1, e_2, \cdots, e_k \in Z_q^*\}$ and $\{\frac{1}{s+e_1}P, \frac{1}{s+e_2}P, \cdots, \frac{1}{s+e_k}P\}$, to compute $\frac{1}{s+e}P$ for some $e \notin \{e_1, e_2, \cdots, e_k\}$.

2.2.Certificateless signature scheme

A certificateless signature scheme is specified by the following six polynomial time algorithms.

Setup. This algorithm takes a security parameter k as input and outputs the system parameters $params$ params and a secret master key $master-key$.

Partial-Private-Key-Extract. This algorithm takes $params$, master-key and a user's identity ID as input. It outputs a partial private key s_{ID} corresponding to the user.

Set-Secret-Value. This algorithm takes the security parameter k and a user's identity ID as input. It outputs the user's secret value x_{ID}.

Set-Private-Key: This algorithm takes $params$, a user's partial private key s_{ID} and his secret value x_{ID} as inputs, and outputs the full private key sk_{ID}.

Set-Public-Key. This algorithm takes a user's secret value x_{ID} as input. It outputs the user's public key pk_{ID}.

Sign. This algorithm takes $params$, a message m, and a user's private key sk_{ID} as input. It outputs a signature σ.

Verify. This algorithm takes $params$, a message m, a user's identity ID, a public key pk_{ID}, and a signature σ as input. It returns 1 means that the signature is accepted. Otherwise, 0 means rejected.

2.3.Security model for certificateless signature scheme

In CLS, as defined in [2], there are two types of adversaries with different capabilities, we assume *Type 1 Adversary*, $\mathcal{A}$ 1 acts as a dishonest user while *Type 2 Adversary*, $\mathcal{A}$ 2 acts as a malicious KGC:

Type 1 Adversary: Adversary $\mathcal{A}$ 1 does not have access to the master key, but $\mathcal{A}$ 1 can replace the public keys of any entity with a value of his choice, since there is no certificate involved in CLS.

Type 2 Adversary: Adversary $\mathcal{A}$ 2 has access to the master key, but cannot replace any user's public key.

Let $\mathcal{A}$ 1 and $\mathcal{A}$ 2 be a Type1 adversary and a Type2 adversary, respectively. We consider two games Game 1 and Game 2 where $\mathcal{A}$ 1 and $\mathcal{A}$ 2 interact with its challenger in these two games, respectively.

Game 1: This is the game where $\mathcal{A}$ 1 interacts with its challenger $\mathcal{C}$:

The challenger $\mathcal{C}$ takes a security parameter l and generate master key and $params$, then sends $params$ to $\mathcal{A}$ 1. $\mathcal{A}$ 1 acts as the following oracle queries:

$Create(ID)$: This allows $\mathcal{A}$ to ask $\mathcal{C}$ to set up a new participant with identity ID. On receiving such a query, $\mathcal{C}$ generates the public/private key pair.

$Public-Key(ID)$: $\mathcal{A}$ can request the public key of a participant whose identity is ID. In response, $\mathcal{C}$ outputs the public key pk_{ID}.

$Partial\text{-}Private\text{-}Key-Extract(ID)$: $\mathcal{A}$ can request the partial private key of a participant whose identity is ID. In response, $\mathcal{C}$ outputs the partial private key s_{ID}.

$Secret\text{-}Key-Extract(ID)$: $\mathcal{A}$ can request the private key of a participant whose identity is ID. In response, $\mathcal{C}$ outputs the private key s_{ID}.

$Public-Key-Replacement(ID, pk'_{ID})$: For a participant whose identity is ID_i, $\mathcal{A}$ can choose a new public key pk'_{ID} and then set pk'_{ID} as the new public key of this participant. $\mathcal{C}$ will record these replacements which will be used later.

$Sign(ID, m)$: When a signing query for an identity ID on some message m is coming, $\mathcal{C}$ uses the private key sk_{ID} corresponding to the identity ID to compute the signature S and sends it to $\mathcal{A}$ 1. If the public key pk_{ID} has been replaced by $\mathcal{A}$ 1, then $\mathcal{C}$ cannot find sk_{ID} and thus the signing oracle's answer may be incorrect. In such case, we assume that $\mathcal{A}$ 1 additionally submits the secret value r' corresponding to the replaced public key sk_{ID} to the signing oracle.

Finally, $\mathcal{A}$ 1 outputs a signature σ on a message m corresponding to a public key pk_{ID^*} for an identity ID^* which is the challenged identity. $\mathcal{A}$ 1 wins the game if the following

conditions hold:

- $Verify(params, ID, m, pk_{ID^*}, \sigma) = 1$
- （ID^*, m）has never been submitted to the oracle $Sign$.
- ID^* has never been submitted to $Partial\text{-}Private\text{-}Key-Extract$ query query.

An adversary $\mathcal{A}1$ is said to be an $(\varepsilon, t, q_c, q_s, q_h)$-forger if it has advantage at least ε in the above game, runs in time at most t, and make at most q_c, q_s and q_h $Create$, $Sign$ and random oracle queries, respectively. A scheme is said to be $(\varepsilon, t, q_c, q_s, q_h)$-secure against $\mathcal{A}1$ in the sense of unforgeable against chosen message attack if no $(\varepsilon, t, q_c, q_s, q_h)$-forger exists.

Game 2: This is a game in which $\mathcal{A}2$ interacts with its challenger $\mathcal{C}$.

Setup: $\mathcal{C}$ runs Setup to generate a master key and $params$. $\mathcal{C}$ gives both $params$ and the master key to $\mathcal{A}$ 2. $\mathcal{C}$ answers $Create(ID)$, $Public-Key(ID)$, $Secret\text{-}Key-Extract(ID)$, $Partial\text{-}Private\text{-}Key-Extract(ID)$ and $Sign(ID, m)$ from $\mathcal{A}2$ like he does in **Game 1**.

Finally, $\mathcal{A}2$ outputs a signature σ on a message m corresponding to a public key pk_{ID^*} for an identity ID^* which is the challenged identity ID. $\mathcal{A}2$ wins the game if the following conditions hold:

- $Verify(params, ID^*, m, pk_{ID^*}, \sigma) = 1$
- （ID^*, m）has never been submitted to the oracle $Sign$.
- ID^* has never been submitted to $Secret\text{-}Key-Extract$ query.

An Type 2 adversary $\mathcal{A}2$ is said to be an $(\varepsilon, t, q_c, q_s, q_h)$-forger if it has advantage at least ε in the above game, runs in time at most t, and make at most q_c, q_s and q_h $Create$, $Sign$ and random oracle queries, respectively. A scheme is said to be $(\varepsilon, t, q_c, q_s, q_h)$-secure against $\mathcal{A}$ 2 in the sense of unforgeable against chosen message attack if no $(\varepsilon, t, q_c, q_s, q_h)$-forger exists.

3. Our scheme

3.1.Scheme Description

A CLS scheme consists of seven algorithms: *Setup*, *Partial-Private-Key-Extract*, *Set-Secret-Value*, *Set-Private-Key*, *Set-Public-Key*, *Sign* and *Verify*. Our scheme also consists of seven algorithms. These algorithms are described as follows.

Setup: This algorithm takes a security parameter l as in put, and returns system parameters and a master key. KGC does the following.

(1) Select a cyclic additive group G_1 of prime order q, a cyclic multiplicative group G_2 of the same order, a generator P of G_1, and a bilinear map $e: G_1 \times G_1 \to G_2$.

(2) Choose a random master-key $x \in Z_q^*$ and set the master public key $P_{pub} = xP$.

(3) Compute $g = e(P, P) \in G_2$.

(3) Choose cryptographic hash functions $H_1: \{0,1\}^* \times G_1 \times G_1 \to \{0,1\}^l$, $H_2: \{0,1\}^* \times \{0,1\}^* \times G_1 \times G_1 \to \{0,1\}^l$.

The system parameters are $params = \{G_1, G_2, e, P, P_{pub}, g, H_1, H_2, l\}$. The master-key is $x \in Z_q^*$.

Set-Secret-Value: The user with identity ID picks randomly $x_{ID} \in Z_n^*$, computes $P_{ID} = x_{ID} \cdot P$ and sets x_{ID} as his secret value.

Partial-Private-Key-Extract: This algorithm takes master key, a user's identifier, P_{ID}, system parameters as input, and returns the user's ID-based private key. With this algorithm, for each user with identifier ID, KGC works as follows.

1) KGC chooses at random $r_{ID} \in Z_n^*$, computes $R_{ID} = r_{ID} \cdot P$ and $h_{ID} = H_1(ID, R_{ID}, P_{ID})$.

2) KGC computes $s_{ID} = r_{ID} + h_{ID} x \bmod n$ and issues $\{s_{ID}, R_{ID}\}$ to the users through secret channel.

The user's s partial private key is the tuple s_{ID} and he can validate her private key by

checking whether the equation $s_{ID} \cdot P = R_{ID} + h_{ID} \cdot P_{pub}$ holds. The private key is valid if the equation holds and vice versa.

Set-Private-Key: The user with identity ID takes the pair $sk_{ID} = (x_{ID}, s_{ID})$ as its private key.

Set-Public-Key: The user with identity ID takes $pk_{ID} = \{P_{ID}, R_{ID}\}$ as its public key.

Sign: This algorithm takes system parameters, user's identity ID, private key $sk_{ID} = (x_{ID}, s_{ID})$, public key $pk_{ID} = (P_{ID}, R_{ID})$ and a message m as inputs, returns a signature of the message m. The user does as follows.

1) Compute $h = H_2(m, ID, P_{ID}, R_{ID})$.

2) Compute $\sigma = \frac{1}{h + x_{ID} + s_{ID}} P$.

3) The resulting signature is σ.

Verify: To verify the signature σ for message m and identity ID, the verifier first computes $h_{ID} = H_1(ID, R_{ID}, P_{ID})$, $h = H_2(m, ID, P_{ID}, R_{ID})$ and then checks whether

$$e(\sigma, hP + P_{ID} + R_{ID} + h_{ID}P) = g \qquad (1)$$

Accept if it is equal. Otherwise reject.

Since $s_{ID} = r_{ID} + h_{ID} x \bmod n$ and $\sigma = \frac{1}{h + x_{ID} + s_{ID}} P$, we have

$$\begin{aligned} & e(\sigma, hP + P_{ID} + R_{ID} + h_{ID}P_{pub}) \\ &= e(\frac{1}{h + x_{ID} + s_{ID}} P, hP + x_{ID}P + r_{ID} \cdot P + h_{ID}xP) \\ &= e(\frac{1}{h + x_{ID} + s_{ID}} P, (h + x_{ID} + r_{ID} + h_{ID}x)P) \qquad (2) \\ &= e(\frac{1}{h + x_{ID} + s_{ID}} P, (h + x_{ID} + s_{ID})P) \\ &= e(P, P) = g \end{aligned}$$

Then the correctness of our scheme is proved.

3.2.Security Analysis

The security proofs of our scheme is similar to the first certificateless short signature scheme[7]. Basically, the main idea of the security proofs given in this section is to have the k-CAA attacker $\mathcal{C}$ simulate the "environment" of the Type 1 and Type 2 attackers $\mathcal{A}1$ and $\mathcal{A}2$

respectively until it can solve k-CCA problem using the ability of $\mathcal{A}1$ and $\mathcal{A}2$.

The following two theorems show that our scheme is secure in the random oracle model, assuming that the k-CAA problem is intractable. We will give the proofs of Theorem 2 and omit the certification process of Theorem 1 due to the similarity of Theorem 2.

Theorem 1. The proposed scheme is $(\varepsilon, t, q_c, q_s, q_h)$-secure against the adversary $\mathcal{A}1$ in the random oracle model, assuming that the (ε', t')-k-CCA assumption holds in G_1, where $t' = t + O(q_c + q_s)S$, $\varepsilon' = (1-\frac{q_h q_c}{n})(1-\frac{q_h^2}{n})\frac{1}{q_c}\varepsilon$ and q_c, q_s, q_h are the number of $Create$, $Sign$ and hashing queries respectively the adversary is allowed to make and S is the time for an scale multiplication operation.

Theorem 2. The proposed scheme is $(\varepsilon, t, q_c, q_s, q_h)$-secure against the adversary $\mathcal{A}2$ in the random oracle model, assuming that the (ε', t')-k-CCA assumption holds in G_1, where $t' = t + O(q_c + q_s)S$, $\varepsilon' = (1-\frac{q_h q_c}{n})(1-\frac{q_h^2}{n})\frac{1}{q_c}\varepsilon$ and q_c, q_s, q_h are the number of $Create$, $Sign$ and hashing queries respectively the adversary is allowed to make and S is the time for an scale multiplication operation.

Proof: Suppose that there is a type 2 Adversar $\mathcal{A}2$ for an adaptively chosen message attack against our scheme. Then, we show how to use the ability of $\mathcal{A}2$ to construct an algorithm $\mathcal{C}$ solving the k-CCA.

Suppose $\mathcal{C}$ is challenged with a k-CCA instance $(P, sP, \frac{1}{s+e_1}P, \frac{1}{s+e_2}P, \cdots, \frac{1}{s+e_k}P)$ and is tasked to compute $\frac{1}{s+e}P$ for some $e \notin \{e_1, e_2, \cdots, e_k\}$. To do so, $\mathcal{C}$ randomly picks a value $x \in Z_n^*$ as the system master key, sets $P_{pub} = x \cdot P$, picks an identity ID^* at random as the challenged ID in this game, and gives the public parameters $\{G_1, G_2, e, P, P_{pub}, g, H_1, H_2, l\}$ and the system master key x to $\mathcal{A}2$. Then $\mathcal{C}$ answers $\mathcal{A}2$'s queries as follows.

$Create(ID)$: $\mathcal{C}$ maintains a hash list L_C of tuple $(ID, R_{ID}, P_{ID}, s_{ID}, x_{ID}, h_{ID})$. If ID is on L_C, then $\mathcal{C}$ response with $(ID, R_{ID}, P_{ID}, s_{ID}, x_{ID}, h_{ID})$. Otherwise, $\mathcal{C}$ simulates the oracle as follows. If $ID = ID^*$, $\mathcal{C}$ chooses $a, b \in Z_n^*$ at random, sets $R_{ID} = aP$,

$h_{ID} = H_1(ID, R_{ID}, P_{ID}) \leftarrow b$, $P_{ID} = sP - R_{ID} - h_{ID}P_{pub}$, $s_{ID} = a + x \cdot h_{ID}$, $x_{ID} \leftarrow \perp$. If $ID \neq ID^*$, $\mathcal{C}$ chooses $a, b, c \in Z_n^*$ at random, sets $R_{ID} = a \cdot P$, $P_{ID} = b \cdot P$, $h_{ID} = H_1(ID, R_{ID}, P_{ID}) \leftarrow c$, $s_{ID} = a + x \cdot h_{ID}$, $x_{ID} = b$. At last $\mathcal{C}$ response with $(ID, R_{ID}, P_{ID}, s_{ID}, x_{ID}, h_{ID})$, inserts $(ID, R_{ID}, P_{ID}, h_{ID})$ into L_{H_1} .

$H_1 - query$: $\mathcal{C}$ maintains a hash list L_{H_1} of tuple $(ID, R_{ID}, P_{ID}, h_{ID})$ as explained below. The list is initially empty. When $\mathcal{A}$ 2 makes a hash oracle query on ID, if the query ID has already appeared on L_{H_1}, then the previously defined value is returned. Otherwise, $\mathcal{C}$ queries $Create(ID)$, gets $(ID, R_{ID}, P_{ID}, s_{ID}, x_{ID}, h_{ID})$ and response with h_{ID}.

$Partial\text{-}Private\text{-}Key - Extract(ID)$: $\mathcal{C}$ looks up the table L_C. If ID is on L_C, then $\mathcal{C}$ response with s_{ID}. Otherwise, $\mathcal{C}$ queries $Create(ID)$, gets $(ID, R_{ID}, P_{ID}, s_{ID}, x_{ID}, h_{ID})$ and response with s_{ID}.

$Public - Key(ID)$: $\mathcal{C}$ looks up the table L_C. If ID is on L_C, then $\mathcal{C}$ response with $pk_{ID} = \{R_{ID}, P_{ID}\}$. Otherwise, $\mathcal{C}$ queries $Create(ID)$, gets $(ID, R_{ID}, P_{ID}, s_{ID}, x_{ID}, h_{ID})$ and response with $pk_{ID} = \{R_{ID}, P_{ID}\}$.

$Secret\text{-}Key - Extract(ID)$: If $ID_i = ID^*$, $\mathcal{C}$ stop the simulation. Otherwise, $\mathcal{C}$ looks up the table L_C. If ID is on L_C, then $\mathcal{C}$ response with x_{ID}. Otherwise, $\mathcal{C}$ queries $Create(ID)$, gets $(ID, R_{ID}, P_{ID}, s_{ID}, x_{ID}, h_{ID})$ and response with x_{ID}.

$H_2 - query$: $\mathcal{C}$ maintains a hash list L_{H_2} of tuple $(m, ID, P_{ID}, R_{ID}, h)$. We assume that $\mathcal{A}$ 2 never repeats a hash query. When $\mathcal{A}$ 2 makes H_2 queries for identity ID on the i th message $M_i = (m_i, ID, P_{ID}, R_{ID})$, if $ID_i \neq ID^*$, $\mathcal{C}$ choose a random number $h \in Z_q^*$, inserts $(m, ID, P_{ID}, R_{ID}, h)$ to L_{H_2}. Otherwise, $\mathcal{C}$ defines $h \leftarrow e_i$ and inserts $(m, ID, P_{ID}, R_{ID}, h)$ to L_{H_2}. At last, $\mathcal{C}$ sends h to $\mathcal{A}$ 2.

$Sign(m_i, ID, P_{ID}, R_{ID})$: We assume that $\mathcal{A}$ 2 never repeats a signature query. When a

signing query on $M_i = (m_i, ID, P_{ID}, R_{ID})$ is coming, if $ID_i \neq ID^*$, $\mathcal{C}$ does as the description of the scheme, since $\mathcal{C}$ knows the secret key $sk_{ID} = \{s_{ID}, x_{ID}\}$. Otherwise, $\mathcal{C}$ outputs $\frac{1}{s+e_i}P$ as the signature.

Finally, $\mathcal{A}$ 2 stops and outputs a signature σ on the message m with respect to the public key pk_{ID} for the identity ID, which satisfies the following equation $Verify(params, ID, m, pk_{ID}, \sigma) = 1$. If $ID \neq ID^*$, $\mathcal{C}$ outputs "failure" and aborts.

Here the hash value of (m, ID, P_{ID}, R_{ID}) is some e and $e \notin \{e_1, e_2, \cdots, e_k\}$. Since $(m, ID, P_{ID}, R_{ID}, \sigma)$ is a valid forgery and it satisfies:

$$\begin{aligned} & e(\sigma, eP + P_{ID} + R_{ID} + h_{ID}P_{pub}) \\ & = e(\sigma, eP + sP - R_{ID} - h_{ID}P_{pub} + R_{ID} + h_{ID}P_{pub}) \\ & = e(\sigma, eP + P_{pub}) = g \end{aligned}$$

So, $\sigma = \frac{1}{s+e}P$. $\mathcal{C}$ outputs (e, σ) as a solution to k-CCA.

Reduction Cost Analysis: The simulation of the $Create$ oracle fails if the random oracle assignment $H_1(ID, R_{ID}, P_{ID})$ causes inconsistency. It happens with probability at most $\frac{q_h}{n}$. Hence, the simulation is successful q_c times with probability at least $(1-\frac{q_h}{n})^{q_c} \geq 1-\frac{q_h q_c}{n}$. The simulation of the H_2 oracle also fails if the random oracle assignment $H_2(m, ID, R_{ID}, P_{ID})$ causes inconsistency. The event happens with probability at most $\frac{q_h}{n}$. Hence, the simulation is successful q_h times with the probability at least $(1-\frac{q_h}{n})^{q_h} \geq 1-\frac{q_h^2}{n}$. In addition, $ID \neq ID^*$ with the probability $\frac{1}{q_c}$. Thus, the overall successful probability is $(1-\frac{q_h q_c}{n})(1-\frac{q_h^2}{n})\frac{1}{q_c}\varepsilon$.

The time complexity of $\mathcal{C}$ is dominated by the exponentiations performed in the $Create$ and $Sign$ queries, which is equal to $t + O(q_c + q_S)S$.

4. Comparison with previous scheme

In this section, we will compare our new scheme with two latest certificateless short signature schemes, i.e. Du et al.'s scheme [9] and Choi et al.'s scheme[11]. For the convenience of evaluating the computational cost, we let s and e denote the scale multiplication operation and the bilinear pairing operation separately. The comparison of our CLS scheme's computation cost and that of other proposed schemes is in Table 1.

Table 1. Comparison of different certificateless short signature schemes

	Sign	Verify	Type 1 attack	Type 2 attack
Du et al.'s scheme[9]	1 s	1 e	Yes	No
Choi et al.'s scheme[11]	3 s	2 s +3e	No	No
Our scheme	1 s	1 e	No	No

From Table 1, we know Du et al.'s scheme [9] suffered from a Type 1 adversary attack although their scheme has almost the same computational cost. Choi et al.'s scheme[11] is secure against both two types adversaries, but their scheme has the worst performance. Given the computational cost of the bilinear pairing operation is about 3 times[13-15] that of the scale multiplication operation, the computational cost of the Sign algorithm and Verify algorithm of our scheme are about 33.3% and 27.28% of Choi et al.'s schemes[11]. Thus our scheme is more practical than the previous schemes.

5. Conclusion

In this paper, we have proposed an efficient certificateless short signature scheme. We also prove the security of the scheme under random oracle model. Compared with previous scheme, the new scheme reduces both the running time of sign algorithm and verify algorithm. Therefore, our scheme is more practical than the previous related schemes for practical application.

6. References

[1]. A. Shamir, Identity-based cryptosystems and signature schemes, Proc. CRYPTO1984, LNCS, vol.196, pp.47–53, 1984.

[2]. S. Al-Riyami, K.G. Paterson, Certificateless public key cryptography, Proceedings of ASIACRYPT 2003, LNCS 2894, Springer-Verlag, 2003, pp. 452 – 473.

[3]. D.H. Yum, P.J. Lee, Generic construction of certificateless signature, ACISP'04, LNCS 3108, Springer, 2004, pp. 200 – 211.

[4]. X. Li, K. Chen, L. Sun, Certificateless Signature and Proxy Signature Schemes from Bilinear Pairings, Lithuanian Mathematical Journal, vol. 45, Springer-Verlag, 2005, pp. 76 – 83.

[5]. Z.F. Zhang, D.S. Wong, J. Xu, et al., Certificateless public-key signature: security model and efficient construction, in: J. Zhou, M. Yung, F. Bao (Eds.), ACNS 2006, LNCS 3989, Springer-Verlag, Berlin, 2006, pp. 293 – 308.

[6]. M.C. Gorantla, A. Saxena, Anefficient certificateless signature scheme, in: Y.Hao, et al., (Eds.), CIS 2005, Part II, LNAI 3802, Springer-Verlag, Berlin, 2005, pp. 110 – 116.

[7]. W.-S. Yap, S.-H. Heng, B.-M. Goi, An efficient certificateless signature scheme, Proc. Of EUC Workshops 2006, LNCS, vol. 4097, 2006, pp. 322 – 331.

[8]. X. Huang, Yi Mu, W. Susilo, D.S. Wong, Certificateless signature revisited, ACISP 2007, LNCS, vol. 4586, Springer-Verlag, 2007, pp. 308 – 322.

[9]. H. Du, Q. Wen, Efficient and provably-secure certificateless short signature scheme from bilinear pairings, Computer Standards & Interfaces 31 (2009) 390 – 394.

[10]. S. Mitsunari, R. Sakai, M. Kasahara, A new traitor tracing, IEICE Trans. E85-A (2) (2002) 481 – 484.

[11]. K. Choi, J. Park, D. Lee, A new provably secure certificateless short signature scheme, Computers and Mathematics with Applications(2011), doi:10.1016/j.camwa.2011.02.003.

[12]. F. Zhang, Re. Safavi-Naini, W. Susilo, An Efficient Signature Scheme from Bilinear Pairings and Its Applications， PKC 2004, LNCS 2947, pp. 277–290, 2004..

[13]. D. He, J. Chen, J. Hu, A pairing-free certificateless authenticated key agreement protocol, Internal Journal of Communication System, DOI: 10.1002/dac.1265.

[14]. D. He, J. Chen, J. Hu, An ID-based proxy signature schemes without bilinear pairings, Annals of Telecommunications, DOI: 10.1007/s12243-011-0244-0.

[15]. X. Cao, W. Kou, A Pairing-free Identity-based Authenticated Key Agreement Protocol with Minimal Message Exchanges, Information Sciences, 10.1016/j.ins.2010.04.002.