New Efficient Certificateless Signature Scheme*

Lei Zhang[1], Futai Zhang[1] and Fangguo Zhang[2]

[1]College of Mathematics and Computer Science,
Nanjing Normal University, P.R. China
[2]Department of Electronics and Communication Engineering,
Sun Yat-Sen University, Guangzhou 510275, P.R.China.
Email: `lei_zhangzl@126.com, zhangfutai@njnu.edu.cn, isszhfg@mail.sysu.edu.cn`

Abstract. In ubiquitous computing environment, how to implement security and trust among the users that connected to a network is a big problem. Digital signature provides authenticity, integrity and non-repudiation to many kinds of applications in ubiquitous computing environment. In this paper, we present a very efficient certificateless signature scheme from bilinear maps. In our scheme, only one paring operation is needed in the signing and verification processes. The security of the new scheme is based on the intractability of the q-Strong Diffie-Hellman (q-SDH) Problem and the Discrete Logarithm Problem. We prove the existential unforgeability of our scheme under adaptively chosen message attack against both types of adversaries in the random oracle model [3].

Keywords: cryptography, certificateless signature scheme, bilinear map, random oracle model.

1 Introduction

To provide the binding between a singer and his public key, the traditional public key signature uses a certificate that is a digitally signed statement issued by the CA. Such certificate can be verified by anyone and guarantees the authenticity of a user's public key. In implementation, the management of public key certificates requires a large amount of computation, storage, and communication cost.

To lower such cost for public key certificate, Shamir [15] proposed another approach named "Identity Based Public Key Cryptography (ID-PKC)" in 1984. In this new approach, a user's public key can be an arbitrary bit string which can represent the user's identity, such as his telephone number or his email address, etc. And the user's corresponding private key is computed by a trusted authority who is referred to as the "Private Key Generator (PKG)" [2,5,12,16]. On input a user's identity and the secret master key owned by PKG, the PKG outputs the user's private key. In this setting, the public key of a user is just his

* Project supported by the nature science foundation of China (No. 60673070), the nature science foundation of Jiangsu province (No. BK2006217), and the open project of the key Lab. on computer networks and information security (Xidian University) of ministry of education of China (No. 20040105).

identity, and no public key certificate is needed. It provides implicit certification of a user's public key based on the fact that only when the user gets a correct private key corresponding to his published identity can he perform some cryptographic operations using his private key. However, there is a basic assumption in identity based cryptosystem, that is the PKG is unconditionally trustable. This is because the PKG knows the private key of every user in the system. So ID-PKC is suffering from the key escrow problem.

To overcome the drawback of key escrow in ID-PKC, Al-Riyami and Paterson [1] proposed a new paradigm called certificateless public key cryptography in 2003. Like ID-PKC, certificateless cryptography does not use public key certificate [1,11,18], it also needs a third party called Key Generation Center (KGC) to help a user to generate his private key. However, the KGC does not have access to a user's full private key. It just generates a user's partial private key from the user's identity as the PKG in ID-PKC does. A user computes his full private key by combining his partial private key and a secret value chosen by himself. The public key of a user is computed from the KGC's public parameters and the secret value of the user, and it is published by the user himself.

Recently, many researchers have been investigating secure and efficient certificateless signature schemes. In their original paper [1], Al-Riyami and Paterson presented a certificateless signature scheme. Huang et al. [9] pointed out a security drawback of the original scheme and proposed a secure one. They also defined the security model of certificateless signature schemes in the same paper. Zhang et al. [21] improved the security model of certificateless signature schemes, and presented a secure certificateless signature scheme. In [18], Yum and Lee presented a generic way to construct certificateless signature schemes, however, Hu et al. [8] pointed out that this construction is insecure and presented a new one. Gorantla and Saxena [7], Yap, Heng, and Goi1 [17] also presented some efficient certificateless signature schemes. Unfortunately, their schemes [7,17] are subject to universal forgery, a type I adversary can forger signatures on any message [6,13,19]. With respect to the efficiency, the previous certificateless signature schemes all involve a relatively large amount of paring computation in the process of verification.

Our contribution. In this paper, we present a new efficient certificateless pairing-based signature scheme, yielding some advantages over previous constructions [7,9,10,17,21] in computational cost. Our signature scheme requires only one pairing operation in the signing and verification phases, so it is much more efficient than the schemes in [7,9,10,17,21]. The security of our scheme is based on the hardness of q-Strong Diffie-Hellman (q-SDH) Problem and the Discrete Logarithm (DL) Problem.

Paper organization. The rest of the paper is organized as follows. Section 2 gives some preliminaries, including bilinear maps, our complexity assumptions, the notions of certificateless signature schemes and their security models. Our new efficient certificateless signature scheme comes in Section 3. In Section 4, we prove the security of our new scheme. The efficiency of our new scheme is compared with some existing certificateless signature schemes in Section 5. Finally, Section 6 comes our conclusion.

2 Preliminaries

2.1 Bilinear Maps and Related Complexity Assumptions

Let G_1 be an additive group of prime order p and G_2 be a multiplicative group of the same order. Let P denote a generator of G_1. A mapping $e : G_1 \times G_1 \longrightarrow G_2$ is called a bilinear mapping if it satisfies the following properties:

1. Bilinear: $e(aP, bQ) = e(P, Q)^{ab}$ for $P, Q \in G_1, a, b \in Z_p^*$.
2. Non-degeneracy: There exists $P, Q \in G_1$ such that $e(P, Q) \neq 1$.
3. Computable: There exists an efficient algorithm to compute $e(P, Q)$ for any $P, Q \in G_1$.

A bilinear pairing instance generator is defined as a probabilistic polynomial time algorithm $\mathcal{IG}$ that takes as input a security parameter l and returns a uniformly random tuple (p, G_1, G_2, e, P) of bilinear parameters, where p is a prime number of size (bit-length) l, G_1 and G_2 are cyclic additive and multiplicative groups of order p respectively, $e : G_1 \times G_1 \longrightarrow G_2$ is a bilinear map, and P is a generator of G_1. For a group G of prime order, we denote the set $G^* = G \setminus \{\mathcal{O}\}$, where $\mathcal{O}$ is the identity element of the group.

Definition 1. Discrete Logarithm (DL) Problem in G_2. *Given a generator g of G_2, and $y \in G_2^*$ to find an integer $a \in Z_p^*$ such that $y = g^a$.*

The DL problem in G_1 can be defined in a similar way.

Definition 2. The q-Strong Diffie-Hellman (q-SDH) problem *in the group G_1 is, given a $(q+1)$-tuple $(P, \alpha P, \alpha^2 P, ..., \alpha^q P)$ as input, finding a pair $(c, \frac{1}{\alpha+c}P)$ with $c \in Z_p^*$.*

Assumption 1. The Discrete Logarithm (DL) Problems in both G_1 and G_2 are intractable.

Assumption 2. The q-SDH Problem in G_1 is intractable.

2.2 Certificateless Signature Schemes

A certificateless signature scheme is defined by seven algorithms: Setup, Partial-Private-Key-Extract, Set-Secret-Value, Set-Private-Key, Set-Public-Key, Sign and Verify. The description of each algorithm is as follows.

– Setup: This algorithm accepts as input a security parameter l and returns a master-key and a list of system parameters params. It also defines the message space $\mathcal{M}$.

- Partial-Private-Key-Extract: This algorithm accepts as input a user's identity ID_i, a parameter list params and a master-key to produce the user's partial private key D_i.
- Set-Secret-Value: This algorithm accepts as input a parameter list params and a user's identity ID_i to produce the secret value x_i for this user.
- Set-Private-Key: This algorithm accepts as input a parameter list params, a user's identity ID_i, his partial private key D_i and secret value x_i to produce a private signing key S_i for this user.
- Set-Public-Key: This algorithm takes as input a parameter list params, a user's identity ID_i and the secret value x_i to produce a public key P_i for this user.
- Sign: This algorithm accepts a message $M \in \mathcal{M}, \mathcal{M}$ is the message space, the signer's identity ID_i and the corresponding public key P_i, a parameter list params and the signing key S_i to generate a signature σ on message M.
- Verify: This algorithm accepts a message M, a signature σ, a parameter list params, the signer's identity ID_i and the corresponding public key P_i to output true if the signature is valid, or $\perp$ otherwise.

2.3 Adversarial Model of Certificateless Signature Schemes

As defined in [1], there are two types of adversary with different capabilities in certificateless signature schemes.

Type I Adversary: This type of adversary $\mathcal{A}_I$ does not have access to the master-key, but $\mathcal{A}_I$ has the ability to replace the public key of any entity with a value of his choice. This is because there is no certificate involved in certificateless signature schemes.

Type II Adversary: This type of adversary $\mathcal{A}_{II}$ has access to the master-key but cannot perform public key replacement.

In this section, firstly we provide a formal definition of existential unforgeability of a certificateless signature scheme against both types of adversaries under chosen message attack. They are defined using the following games between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}_I$ or $\mathcal{A}_{II}$.

Game 1 (for *Type I Adversary*)

- Setup: $\mathcal{C}$ runs the Setup algorithm, takes as input a security parameter l to obtain the master-key and the system parameter list params. $\mathcal{C}$ then sends params to the adversary $\mathcal{A}_I$.
- Partial-Private-Key Queries $\mathrm{PPK}(ID_i)$: $\mathcal{A}_I$ can request the partial private key of any user with identity ID_i. In respond, $\mathcal{C}$ replies the partial private key D_i of the user.
- Public-Key Queries $\mathrm{PK}(ID_i)$: $\mathcal{A}_I$ can request the public key of a user with identity ID_i. In respond, $\mathcal{C}$ outputs the public key P_i.
- Private-Key Queries $\mathrm{Pr}(ID_i)$: $\mathcal{A}_I$ can request the private key of a user with identity ID_i. In respond, $\mathcal{C}$ outputs the private key S_i.

- Public-Key-Replacement Queries $\mathrm{PKR}(ID_i,\ P_i')$: This query is to replace the public key P_i for an identity ID_i with a new value P_i'. On receiving such a query, $\mathcal{C}$ updates the public key to the new value P_i'.
- Sign Queries $\mathrm{S}(M, ID_i, P_i)$: $\mathcal{A}_I$ can request a user's (whose identity is ID_i) signature on a message M. On receiving a query $\mathrm{S}(M, ID_i, P_i)$, $\mathcal{C}$ generates a signature σ on message M and replies with (M, σ, ID_i, P_i).
- Output: This procedure contains three steps.
  Step 1: Select target identity: $\mathcal{A}_I$ selects a target identity ID^*, chooses a new public key P_{ID^*} for this identity. He Submits (ID^*, P_{ID^*}) to $\mathcal{C}$.
  Step 2: Further queries: $\mathcal{A}_I$ can make more Partial-Private-Key, Public-Key, Private-Key, Public-Key-Replacement and Sign Queries.
  Step 3: Forge: $\mathcal{A}_I$ outputs a tuple $(M^*, \sigma^*, ID^*, P_{ID^*})$. This tuple must satisfy the following requirements:
  1. σ^* is a valid signature on message M^* for user ID^* under public key P_{ID^*} chosen by $\mathcal{A}_I$.
  2. $\mathcal{A}_I$ has never asked the partial private key or private key of the user whose identity is ID^*.
  3. $\mathrm{S}(M^*, ID^*, P_{ID^*})$ has never been queried during the Sign Queries.

Definition 3. *A certificateless signature scheme is existentially unforgeable against Type I adversary under adaptively chosen-message attacks iff the probability of success of any polynomially bounded Type I adversary in the above game is negligible.*

Game 2 (for *Type II Adversary*)

- Setup: $\mathcal{C}$ runs the Setup algorithm, takes as input a security parameter l to obtain the system parameter list params and also the system's master-key. $\mathcal{C}$ then sends params and master-key to the adversary $\mathcal{A}_{II}$.
- Public-Key Queries $\mathrm{PK}(ID_i)$: $\mathcal{A}_{II}$ can request a user's (whose identity is ID_i) public key. On receiving a query $\mathrm{PK}(ID_i)$. $\mathcal{C}$ replies the public key P_i.
- Private-Key Queries $\mathrm{Pr}(ID_i)$: $\mathcal{A}_{II}$ can request the private key of a user with identity ID_i. In respond, $\mathcal{C}$ outputs the private key S_i.
- Sign Queries $\mathrm{S}(M, ID_i, P_i)$: $\mathcal{A}_{II}$ can request a user's (whose identity is ID_i) signature on a message M. On receiving a query $\mathrm{S}(M, ID_i, P_i)$, $\mathcal{C}$ replies with a signature σ on message M for the user with identity ID_i under public key P_i.
- Output: This procedure contains three steps.
  Step 1: Select target identity: $\mathcal{A}_{II}$ selects a target identity ID^* whose public key has been asked during Public-Key Queries. He Submits (ID^*, P_{ID^*}) to $\mathcal{C}$.
  Step 2: Further queries: $\mathcal{A}_{II}$ can make more Public-Key, Private-Key and Sign Queries.
  Step 3: Forge: $\mathcal{A}_{II}$ outputs a tuple $(M^*, \sigma^*, ID^*, P_{ID^*})$. This tuple must satisfy the following requirements:
  1. This signature is a valid one, i.e. it passes the verification algorithm with respect to the identity ID^* under the public key P_{ID^*}.
  2. $\mathcal{A}_{II}$ has never asked the private key of the user with identity ID^*.

3. S(M^*, ID^*, P_{ID^*}) has never been queried during the Sign Queries.

Definition 4. *A certificateless signature scheme is existentially unforgeable against Type II adversary under adaptively chosen-message attacks iff the probability of success of any polynomially bounded Type II adversary in the above game is negligible.*

Definition 5. *A certificateless signature scheme is existentially unforgeable under adaptively chosen-message attacks iff it is existentially unforgeable against both types of adversaries.*

3 Our Scheme

In this section, we present an efficient certificateless signature scheme. The construction is as follows.

- Setup: When input a security parameter l, this algorithm runs as follows.
  1. Run $\mathcal{IG}$ on input 1^l to generate (p, G_1, G_2, e, P), set $g = e(P, P)$.
  2. Choose a random master-key $s \in Z_p^*$ and set $P_0 = sP$.
  3. Choose cryptographic hash functions $H_1 : \{0,1\}^* \longrightarrow Z_p^*$ and $H_2 : \{0,1\}^n \times G_2 \times G_2 \times G_2 \longrightarrow Z_p^*$, where n denote the bit-length of plaintexts.

  The system parameters params=$(G_1, G_2, e, n, P, P_0, g, H_1, H_2)$. The master-key is $s \in Z_p^*$. The message space is $\mathcal{M}= \{0,1\}^n$.
- Partial-Private-Key-Extract [20]: This algorithm accepts an identity $ID_i \in \{0,1\}^*$ of a user and generates the partial private key for the user as follows.
  1. Compute $y_i = H_1(ID_i)$.
  2. Output the partial private key $D_i = \frac{1}{s+y_i}P$.
- Set-Secret-Value: This algorithm takes as input params and a user's identity ID_i. It selects a random $x_i \in Z_p^*$ and outputs x_i as the user's secret value.
- Set-Private-Key: This algorithm takes as input params, a user's identity ID_i, the user's partial private key D_i and secret value $x_i \in Z_p^*$. The output of the algorithm is the private key $S_i = (x_i, D_i)$.
- Set-Public-Key: This algorithm accepts params, a user's identity ID_i and secret value $x_i \in Z_p^*$ to produce the user's public key $P_i = g^{x_i}$.
- Sign: To sign a message $M \in \mathcal{M}$ using the private key S_i, a signer with identity ID_i and corresponding public key P_i, performs the following steps.
  1. Select random $r_1, r_2 \in Z_p^*$.
  2. Compute $R = g^{r_1}, R' = g^{r_2}$, set $v = H_2(M, R, R', P_i)$.
  3. Compute $U = (x_i v + r_1)D_i, w = x_i v + r_2$.
  4. Output (U, v, w) as the signature on M.
- Verify: To verify a signature (U, v, w) on a message M for an identity ID_i under public key P_i, the verifier performs the following steps.
  1. Compute $R = e(U, P_0 + H_1(ID_i)P)P_i^{-v}, R' = g^w P_i^{-v}$.
  2. Verify $v \overset{?}{=} H_2(M, R, R', P_i)$ holds with equality.

  If it does, output true. Otherwise, output $\perp$.

4 Security Proof

Assuming that the q-SDH problem in G_1 and DL problems in both G_1 and G_2 are hard, we now prove the security of the above signature scheme.

Theorem 1. *Our scheme is unforgeable against type I adversary in the random oracle model assuming the q-SDH problem in G_1 is intractable.*

Proof. Let $\mathcal{C}$ be a q-SDH problem attacker, $\mathcal{A}$ is a type I adversary who interacts with $\mathcal{C}$ following Game 1. We take hash functions H_1 and H_2 as random oracles. Assume that $\mathcal{A}$'s target identity is ID^*, and he can forge a valid signature on a message M^* for the identity ID^*.

$\mathcal{C}$ is given $(P, \alpha P, \alpha^2 P, ..., \alpha^q P)$ as an input to the q-SDH problem and aims to find a pair $(c, \frac{1}{\alpha+c}P)$. In Setup phase, it selects a generator $P' \in G_1$ such that it knows $q-1$ pairs $(h_i, \frac{1}{\alpha+h_i}P')$ for random $h_1, ..., h_{q-1} \in Z_p^*$. To do so,

1. It picks random $h_1, ..., h_{p-1} \in Z_p^*$ and expands $f(z) = \prod_{i=1}^{q-1}(z+h_i)$ to obtain $c_0, ..., c_{q-1} \in Z_p^*$ so that $f(z) = \sum_{i=0}^{q-1} c_i z^i$.
2. It sets $P' = \sum_{i=0}^{q-1} c_i(\alpha^i P) = f(\alpha)P$, the public key P_0' is fixed to $P_0' = \sum_{i=1}^{q} c_{i-1}(\alpha^i P) = \alpha P'$ although $\mathcal{C}$ does not know α.
3. For $1 \leq i \leq q-1$, $\mathcal{C}$ expands $f_i(z) = f(z)/(z+h_i) = \sum_{i=0}^{q-2} d_i z^i$ and gets $\sum_{i=0}^{q-2} d_i(\alpha^i P) = f_i(\alpha)P = \frac{f(\alpha)}{\alpha+h_i}P = \frac{1}{\alpha+h_i}P'$. The pairs $(h_i, \frac{1}{\alpha+h_i}P')$ are computed.

We let $g' = e(P', P')$, the params given to $\mathcal{A}$ is $(G_1, G_2, e, n, P', P_0', g', H_1, H_2)$, which has the correct distribution.

H_1 queries: For simplicity, we assume that the queries to H_1 are distinct. When $\mathcal{A}$ issues a query ID_i to H_1, $\mathcal{C}$ replies h_i which is previously selected and increments i. At some point, $\mathcal{A}$ uniformly chooses an identity ID^* and submits it to $\mathcal{C}$. In response, $\mathcal{C}$ replies $c \in Z_p^*$ which is randomly selected.

H_2 queries: It can be naturally simulated. Namely, whenever $\mathcal{A}$ issues a query (M_i, R_i, R_i', P_i) to H_2, $\mathcal{C}$ picks $v_i \in Z_p^*$ at random and returns v_i as answer.

Partial-Private-Key Queries: $\mathcal{C}$ maintains a initially empty list K^{list}. When $\mathcal{A}$ issues a query PPK(ID^*), $\mathcal{C}$ aborts. While $\mathcal{A}$ issues a query PPK(ID_i) where $ID_i \in \{ID_1, ..., ID_{q-1}\}$, the same answer from K^{list} will be given if the request has been asked before; otherwise, $\mathcal{C}$ does as follows

1. If there's a tuple (ID_i, D_i, x_i, P_i) which is indexed by ID_i is found on K^{list}, then $\mathcal{C}$ sets $D_i = \frac{1}{\alpha+h_i}P'$ which is previously computed, returns D_i as answer.
2. Otherwise, $\mathcal{C}$ sets $D_i = \frac{1}{\alpha+h_i}P'$ which is previously computed, returns D_i as answer and adds (ID_i, D_i, x_i, P_i) to K^{list}.

Public-Key Queries: When $\mathcal{A}$ issues a query PK(ID) where $ID \in \{ID_1, ..., ID_{q-1}, ID^*\}$, the current public key relates to ID from K^{list} will be given if the request has been asked before; otherwise, $\mathcal{C}$ does as follows

1. If the query is on ID^*, when there's a tuple (ID^*, D^*, x^*, P^*) which is indexed by ID^* is found on K^{list}, $\mathcal{C}$ selects a random $x^* \in Z_p^*$, sets the public key $P^* = g'^{x^*}$, returns P^* as answer and updates(ID^*, D^*, x^*, P^*) to the new value; while no such a tuple matches, $\mathcal{C}$ sets $D^* = \perp$, selects a random $x^* \in Z_p^*$, computes the public key $P^* = g'^{x^*}$, returns P^* as answer and adds (ID^*, D^*, x^*, P^*) to K^{list}.
2. Otherwise, the query is on $ID_i \in \{ID_1, ..., ID_{q-1}\}$. When there's a tuple (ID_i, D_i, x_i, P_i) which is indexed by ID_i is found on K^{list}, $\mathcal{C}$ selects a random $x_i \in Z_p^*$, sets the public key $P_i = g'^{x_i}$, returns P_i as answer and updates (ID_i, D_i, x_i, P_i) to the new value; while no such a tuple matches, $\mathcal{C}$ selects a random $x_i \in Z_p^*$, computes the public key $P_i = g'^{x_i}$, returns P_i as answer and adds (ID_i, D_i, x_i, P_i) to K^{list}.

Private-Key Queries: When $\mathcal{A}$ issues a query $\Pr(ID)$ where $ID \in \{ID_1, ..., ID_{q-1}, ID^*\}$, if $ID = ID^*$, $\mathcal{C}$ aborts; else if $\mathcal{A}$ has ever made an Public-Key-Replacement query on ID, $\mathcal{C}$ returns $\perp$; otherwise, $\mathcal{C}$ first makes Partial-Private-Key and Public-Key Queries on ID, if $\mathcal{C}$ does not abort, then returns the private key of the user whose identity is ID.

Public-Key-Replacement Queries: $\mathcal{A}$ can replace any user's public key as stated in Game 1.

On receive a Sign query $\mathrm{S}(M, ID, P_{ID})$, where $ID \in \{ID_1, ..., ID_{q-1}, ID^*\}$ and P_{ID} denotes the current public key of the user whose identity is ID, $\mathcal{C}$ creates a signature as follows

1. Pick $U_* \in G_1$, $v_* \in Z_p^*$ and $w_* \in Z_p$ at random.
2. Compute $R_* = e(U_*, P_0' + H_1(ID)P')P_{ID}^{-v_*}, R_*' = g'^{w_*}P_{ID}^{-v_*}$.
3. Set $H_2(M, R_*, R_*', P_{ID}) = v_*$.
4. Return $(M, \sigma = (U_*, v_*, w_*), ID, P_{ID})$ as answer.

Note that $\mathcal{A}$ (everyone) can verify $\sigma = (U_*, v_*, w_*)$ is a valid signature on message M for identity ID under public key P_{ID}.

The next step of the simulation is to apply the 'forking' technique formalized in [14]: Let ID^* is the target identity that $\mathcal{A}$ has chosen. Suppose $(M^*, (U, v, w), ID^*, P_{ID^*})$ be a forgery that output by $\mathcal{A}$ at the end of the attack. Note that if $\mathcal{A}$ does not output ID^* as a part of the forgery, $\mathcal{C}$ just aborts the simulation. $\mathcal{C}$ then replays $\mathcal{A}$ with the same random tape but different choice of the hash function H_2' to get another forgery $(M^*, (U', v', w'), ID^*, P_{ID^*})$. From these two forgeries, $\mathcal{C}$ obtains

$$R = e(U, P_0' + cP')P_{ID^*}^{-v}, R' = g'^{w}P_{ID^*}^{-v}$$

and

$$R = e(U', P_0' + cP')P_{ID^*}^{-v'}, R' = g'^{w'}P_{ID^*}^{-v'}$$

Since (U, v, w) and (U', v', w') are valid signatures on M^*, $\mathcal{C}$ consequently obtains the following (Here we let $P_{ID^*} = g'^{a}$):

$$\begin{aligned} g'^{w} P_{ID^*}^{-v} &= g'^{w'} P_{ID^*}^{-v'} \\ g'^{w} g'^{-av} &= g'^{w'} g'^{-av'} \\ g'^{a} &= g'^{(v-v')^{-1}(w-w')} \end{aligned}$$

Since $\mathcal{C}$ has the knowledge of (v, v', w, w'), he can compute $a = (v-v')^{-1}(w-w')$.

$\mathcal{C}$ also obtains the following:

$$\begin{aligned} e(U, (\alpha+c)P')P_{ID^*}^{-v} &= e(U', (\alpha+c)P')P_{ID^*}^{-v'} \\ e(U, (\alpha+c)P')e(P', P')^{-av} &= e(U', (\alpha+c)P')e(P', P')^{-av'} \\ e((\alpha+c)U - avP', P') &= e((\alpha+c)U' - av'P', P') \end{aligned}$$

From the last equation, $\mathcal{C}$ has the following

$$\begin{aligned} (\alpha+c)U - avP' &= (\alpha+c)U' - av'P' \\ (\alpha+c)(U-U') &= a(v-v')P' \end{aligned}$$

Since $\mathcal{C}$ has the knowledge of (v, v', a, U, U'), he can compute

$$\frac{1}{\alpha+c}P' = a^{-1}(v-v')^{-1}(U-U')$$

From $\frac{1}{\alpha+c}P'$, $\mathcal{C}$ can proceed as in [2,4] to extract $\frac{1}{\alpha+c}P$: It first obtains $\gamma_{-1}, \gamma_0, ..., \gamma_{q-2} \in Z_p^*$ for which $f(z)/(z+h) = \gamma_{-1}/(z+h) + \sum_{i=0}^{q-2} \gamma_i z^i$ and eventually computes

$$\frac{1}{\alpha+c}P = \frac{1}{\gamma_{-1}}\left[\frac{1}{\alpha+c}P' - \sum_{i=0}^{q-2}\gamma_i \alpha^i P\right]$$

So $\mathcal{C}$ has successfully obtains the solution of q-SDH problem. By now, we obtain a contradiction and hence, complete the proof.

Theorem 2. *Our scheme is existentially unforgeable against the type II adversary in the random oracle model assuming the DL problem is intractable.*

Proof. Let $\mathcal{A}$ be our type II adversary. $\mathcal{A}$ has access to the master-key, but cannot perform any public key replacement. $\mathcal{C}$ is given an instance (g, g^a) of the DL problem in G_2. We will show how can $\mathcal{C}$ solve the DL problem (i.e. to compute a) using $\mathcal{A}$'s capability as follows.

Firstly, $\mathcal{C}$ generates the KGC's master-key $s \in Z_p^*$ and the system parameters params=$(G_1, G_2, e, n, P, P_0, g, H_1, H_2)$. Then $\mathcal{A}$ is provided with params and the master-key s. Since $\mathcal{A}$ has access to the master-key, he can do Partial-Private-Key-Extract himself.

Suppose that $\mathcal{A}$ can forge a valid signature on message M^* for identity ID^* under public key P_{ID^*}. $\mathcal{C}$ sets ID^*'s public key as $P_{ID^*} = g^a$ for some unknown a. When $\mathcal{A}$ issues an H_1 query on ID_i, $\mathcal{C}$ picks a random $h_i \in Z_p^*$ and returns as answer. While for an H_2 query on (M_i, R_i, R_i', P_i), $\mathcal{C}$ picks a random $v_i \in Z_p^*$ and returns as answer. When $\mathcal{A}$ issues a public key query on an identity $ID_i \neq ID^*$,

$\mathcal{C}$ picks a random $x_i \in Z_p^*$ as ID_i's secret value, computes $P_i = g^{x_i}$, returns P_i as answer and adds the tuple (ID_i, D_i, x_i, P_i) to K^{list} which is initially empty (where $D_i = \frac{1}{s+H_1(ID_i)}P$); otherwise, returns $P_{ID^*} = g^a$. Whenever $\mathcal{A}$ submits a private key query on ID_i, if $ID_i = ID^*$, $\mathcal{C}$ aborts; otherwise $ID_i \neq ID^*$, if the query PK(ID_i) has not been queried, he first makes PK(ID_i), eventually returns (x_i, D_i) as answer. To answer a Sign query, $\mathcal{C}$ replies with a valid signature if the query is not S(M^*, ID^*, P_{ID^*}) (the simulation is the same as mentioned in the proof process of Theorem 1); otherwise, he aborts. Suppose $\mathcal{A}$ eventually outputs a valid signature (U, v, w) on message M^* under identity ID^* and public key P_{ID^*}. Applying the forking technique, a set of two forged signatures (U, v, w) and (U', v', w') on the same message M^* for identity ID^* under public key P_{ID^*} will be obtained. When this happens, $\mathcal{C}$ gets

$$R = e(U, P_0 + H_1(ID^*)P)P_{ID^*}^{-v}, R' = g^w P_{ID^*}^{-v}$$

and

$$R = e(U', P_0 + H_1(ID^*)P)P_{ID^*}^{-v'}, R' = g^{w'} P_{ID^*}^{-v'}$$

Since (U, v, w) and (U', v', w') are valid signatures on M^*, $\mathcal{C}$ consequently obtains the following

$$\begin{aligned} g^w P_{ID^*}^{-v} &= g^{w'} P_{ID^*}^{-v'} \\ g^w g^{-av} &= g^{w'} g^{-av'} \\ g^a &= g^{(v-v')^{-1}(w-w')} \end{aligned}$$

Because $\mathcal{C}$ has the knowledge of (v, v', w, w'), he can compute $a = (v-v')^{-1}(w-w')$. And hence, $\mathcal{C}$ has successfully obtains the solution of DL problem.

5 Efficiency

Table 1 gives a comparison of computational efforts required for our scheme with that of the signature schemes in [7,9,10,17,21] in the Sign and Verify algorithms. Here we only consider the costly operations which defined below, and we omit the computational effort of the hash operation $H(ID)$ in the Sign algorithm, since it can be computed only once.

Table 1. Comparison of Computational Efforts

Schemes	Sign	Verify
Scheme in [7]	$2S$	$3P + 1S + 1H$
Scheme in [9]	$2P + 3S$	$4P + 1H + 1E$
Scheme in [10]	$2S + 1H$	$4P + 1S + 2H$
Scheme in [17]	$2S$	$2P + 1S + 1H$
Scheme in [21]	$3S + 2H$	$4P + 3H$
Our Scheme	$1S + 2E$	$1P + 1S + 2E$

P: Pairing Operation S: Scalar Multiplication in G_1
H: MapToPoint Hash E: Exponentiation in G_2

Our Sign algorithm requires no pairing operation and two exponentiation operations in G_2. Our Verify algorithm requires only one pairing operation, much less than it is required in the Verify algorithms of the other schemes [7,9,10,17,21].

6 Conclusion

It is interesting to investigate secure and efficient certificateless signature schemes. In this paper, we have proposed a secure certificateless signature scheme. The scheme is constructed from bilinear maps. An advantage of our new scheme over the other existing certificateless signature schemes is its efficiency in computation. The total number of pairing operations in the signing and verification processes of our new scheme is one. This is probably the best to achieve in pairing based signature schemes. The proofs of the existential unforgeability of our new scheme under adaptively chosen message attack for both types of adversaries are given as well.

References

1. S. Al-Riyami and K. Paterson, Certificateless public key cryptography, Asiacrypt 2003, LNCS, vol.2894, pp.452-473, Springer-Verlag, 2003.
2. P. Barreto, B. Libert, N. McCullagh, and J. Quisquater, Efficient and provably-secure identity-based signatures and signcryption from bilinear maps, ASIACRYPT 2005, LNCS, vol.3788, pp.515-532, Springer-Verlag, 2005.
3. M. Bellare and P. Rogaway, Random oracles are practical: A paradigm for designing efficient protocols, ACM CCCS '93, pp.62-73, 1993.
4. D. Boneh and X. Boyen, Short signatures without random oracles, Eurocrypt'04, LNCS, vol.3027, pp.56-73, Springer-Verlag, 2004.
5. D. Boneh and F. Franklin, Identity-based encryption from the Weil pairing, Crypto 2001, LNCS, vol.2139, pp.213-229, Springer-Verlag, 2001.
6. X. Cao, K. Paterson, and W. Kou, An attack on a certificateless signature scheme, Cryptology ePrint Archive, Report 2006/367, 2006.
7. M. Gorantla,A. Saxena, An efficient certificateless signature scheme, CIS 2005, LNCS, vol.3802, pp.110-116, 2005.
8. B. Hu, D. Wong, Z. Zhang and X. Deng, Key replacement attack against a generic construction of certificateless signature, ACISP 2006, LNCS, vol.4058, pp.235-346, Springer-Verlag, 2006.
9. X. Huang, W. Susilo, Y. Mu and F. Zhang, On the security of a certificateless signature scheme, CANS 2005, LNCS, vol.3810, pp.13-25, Springer-Verlag, 2005.
10. X. Li, K. Chen and L. Sun. Certificateless signature and proxy signature schemes from bilinear pairings. *Lithuanian Mathematical Journal*, vol. 45, pages 76-83, Springer-Verlag, 2005.
11. B. Libert and J. Quisquater, On constructing certificateless cryptosystems from identity based encryption, PKC 2006, LNCS, vol.3958, pp.474-490, Springer-Verlag, 2006.
12. Y. Mu and W. Susilo, Identity-based instantaneous broadcast system in mobile adhoc networks, the 2004 International Workshop on Mobile Systems, E-commerce and Agent Technology, USA, 2004, pp35-40.

13. J. Park, An attack on the certificateless signature scheme from EUC Workshops 2006, Cryptology ePrint Archive, Report 2006/442, 2006.
14. D. Pointcheval and J. Stern, Security proofs for signature schemes, Eurocrypt 1996, LNCS, vol.1070, pp.387-398, Springer-Verlag, 1996.
15. A. Shamir, Identity based cryptosystems and signature schemes, Advances in Cryptology-Crypto'84, LNCS, vol.196, pp.47-53, Springer-Verlag, 1984.
16. W. Susilo, F. Zhang, and Y. Mu, Identity-based strong designated verifier signature schemes, ACISP 2004, LNCS, vol.3108, pp.313-324, Springer-Verlag, 2004.
17. W. Yap, S. Heng, and B. Goi1, An efficient certificateless signature scheme, EUC Workshops 2006, LNCS, vol.4097, pp.322-331, Springer-Verlag, 2006.
18. D. Yum and P. Lee, Generic construction of certificateless signature, ACISP 2004, LNCS, vol.3108, pp.200-211, Springer-Verlag, 2004.
19. Z. Zhang and D. Feng, Key replacement attack on a certificateless signature scheme. Cryptology ePrint Archive, Report 2006/453, 2006.
20. F. Zhang, R. Safavi-Naini, W. Susilo, An efficient signature scheme from bilinear pairings and its applications. PKC 2004, LNCS, vol.2947, pp.277-290, Springer-Verlag, 2004.
21. Z. Zhang, D. Wong, J. Xu and D. Feng, Certificateless public-key signature: security model and efficient construction, ACNS 2006, LNCS, vol.3989, pp.293-308, Springer-Verlag, 2006.