2009

A concrete certificateless signature scheme without pairings

Aijun Ge
Zhengzhou Information Science and Technology Institute, geaijun@163.com

Shaozhen Chen
Zhengzhou Information Science and Technology Institute, chenshaozhen@vip.sina.com

Xinyi Huang
University of Wollongong, xh068@uow.edu.au

Publication Details
A concrete certificateless signature scheme without pairings

Abstract
Certificateless public key cryptography was introduced to avoid the inherent key escrow problem in identity-based cryptography, and eliminate the use of certificates in traditional PKI. Most cryptographic schemes in certificateless cryptography are built from bilinear mappings on elliptic curves which need costly operations. Despite the investigation of certificateless public key encryption without pairings, certificateless signature without pairings received much less attention than what it deserves. In this paper, we present a concrete pairing-free certificateless signature scheme for the first time. Our scheme is more computationally efficient than others built from pairings. The new scheme is provably secure in the random oracle model assuming the hardness of discrete logarithm problem.

Disciplines
Physical Sciences and Mathematics

Publication Details
A Concrete Certificateless Signature Scheme without Pairings

Aijun Ge, Shaozhen Chen
Department of Applied Mathematics
Zhengzhou Information Science and Technology Institute
Zhengzhou, Henan 450002, China
Email: geaijun@163.com, chenshaozhen@vip.sina.com

Xinyi Huang
School of Computer Science and Software Engineering
University of Wollongong
Wollongong, NSW 2500, Australia
Email: xyhuang81@gmail.com

Abstract—Certificateless public key cryptography was introduced to avoid the inherent key escrow problem in identity-based cryptography, and eliminate the use of certificates in traditional PKI. Most cryptographic schemes in certificateless cryptography are built from bilinear mappings on elliptic curves which need costly operations. Despite the investigation of certificateless public key encryption without pairings, certificateless signature without pairings received much less attention than what it deserves. In this paper, we present a concrete pairing-free certificateless signature scheme for the first time. Our scheme is more computationally efficient than others built from pairings. The new scheme is provably secure in the random oracle model assuming the hardness of discrete logarithm problem.

Keywords—certificateless signature; bilinear pairing; discrete logarithm problem; random oracle model; strongly secure

I. INTRODUCTION

In Asiacrypt 2003, Al-Riyami and Paterson proposed the notion of “Certificateless Public Key Cryptography” [1], whose original motivation is to find a public key system that does not use certificates, and at the same time does not have the key escrow problem. In certificateless cryptography, each user has two secrets, namely a secret value and a partial private key. The former is generated by the entity itself, and the latter is produced by a third party called as the Key Generating Center (KGC), who holds a master key. Decrypting or signing requires both secrets. As KGC does not know the secret value generated by the user, the key escrow problem is eliminated. The corresponding public key is generated by using the secret value and (optional) the partial private key. In certificateless cryptography, one’s public key could be made available to other users by transmitting it along with messages (for example, in a signing application) or by placing it in a public directory (this would be more appropriate for an encryption setting).

Since the introduction of certificateless cryptography, a lot of schemes have been proposed so far (e.g., [4], [5], [6], [8], [9], [11]). However, as pointed out by Baek et al. [2], certificateless encryption schemes (and certificateless signature schemes) have been constructed within the framework of ID-based encryption proposed by Boneh and Franklin [3]. As a result, most certificateless cryptography schemes are based on bilinear mappings on elliptic curves, which are used to construct identity-based encryption and require heavy computational cost. Being aware of this, Baek et al. [2] proposed the first certificateless encryption scheme which does not depend on bilinear mappings. Sun et al. [12] improved their scheme and proposed a strongly secure certificateless encryption without pairings. Both schemes [2], [12] are more computationally efficient than others from bilinear mappings.

To the best of our knowledge, all concrete constructions of certificateless signatures in the literature are built from bilinear mappings. In this paper, we present the first concrete efficient certificateless signature scheme without pairings, and prove its security in the random oracle model. The security of our scheme can be reduced to discrete logarithm problem in finite fields. Our work is motivated by certificateless encryption schemes proposed in [2], [12]. Namely, by incorporating Schnorr signature [10] nontrivially, we obtain a certificateless signature scheme without pairings.

II. PRELIMINARIES

This section reviews definitions of certificateless signatures and complexity assumptions associated with our scheme.

A. Syntax of Certificateless Signature Scheme

Definition 1: A certificateless signature scheme is made up of seven algorithms: Setup, Partial-Key-Extract, Set-Secret-Value, Set-Public-Key, Set-Private-Key, Sign, Verify. For a fixed security parameter \( k \), these algorithms work as follows:

- **Setup(\( k \)).** This algorithm takes a security parameter \( k \) as input and returns the master secret key \( msk \), the master public key \( mpk \) and a list of public system parameters \( params \).
- **Partial-Key-Extract(\( params, ID, msk \)).** This algorithm takes system parameters \( params \), the master secret key \( msk \) and a user’s identity \( ID \) as inputs, and returns a partial private key \( \mathcal{D}_{ID} \) and (optional) a partial public key \( \mathcal{P}_{ID} \) corresponding to the user with the identity \( ID \).
- **Set-Secret-Value(\( params, msk \)).** This algorithm takes system parameters \( params \) and the master public key \( mpk \) as inputs, and returns a secret value \( s_{ID} \).
- **Set-Public-Key(\( params, mpk, ID, \mathcal{P}_{ID}, s_{ID} \)).** This algorithm takes system parameters \( params \), the
master public key $mpk$, the user's identity ID, ID's partial public key $P_{ID}$ and secret value $s_{ID}$ as inputs, and returns the public key $PK_{ID}$.

- **Set-Private-Key**($params, D_{ID}, s_{ID}$).
  This algorithm takes system parameters $params$, a user's partial private key $D_{ID}$ and his/her secret value $s_{ID}$ as inputs, and returns the private key $SK_{ID}$.

- **Sign**($params, mpk, ID, SK_{ID}, m$).
  This algorithm takes system parameters $params$, the master public key $mpk$, the user's identity ID, his/her private key $SK_{ID}$ and the message $m$ to be signed as inputs, and returns a certificateless signature $\sigma$.

- **Verify**($params, mpk, ID, PK_{ID}, m, \text{Sign}(params, mpk, ID, SK_{ID}, m)$) = valid.

### Completeness
For any correctly generated key pair $(SK_{ID}, PK_{ID})$.

#### B. Security Model of Certificateless Signatures

As there is no certificate to authenticate a user’s public key, it is reasonable to assume that an adversary can replace the user's public key with any value of its choice. Thus, two types of adversaries have been defined in certificateless cryptography [1]. A Type I adversary can replace any user’s public key but does not have the partial private key of the target user, while a Type II adversary simulates a dishonest KGC who has the knowledge of the master secret key (and thus the partial private keys of all users), but is not allowed to replace the target user’s public key.

For the security model of our certificateless signature scheme, we consider the strongest Type I adversaries defined in [5]: Super Type I adversary $A_{II}$, which is given as much power as possible. $A_{II}$ can obtain some message/signature pairs which are valid under the public key chosen by itself without providing the corresponding secret value.

For Type II adversary, we also consider the strongest adversary model “Super Type II adversary” defined in [5], which is given as much power as possible. $A_{II}$ is allowed to obtain some message/signature pairs which are valid under the public key chosen by itself without providing the corresponding secret value. Note that $A_{II}$ is not allowed to replace the target user’s public key.

**Definition 2**: Let $\text{Succ}_{cma, cida}^{\text{ema, cida}}$ be the success probability of a Super Type I adaptively chosen message and chosen identity adversary $A_{I}$, a certificateless signature scheme is secure against Super Type I adversary $A_{I}$ if $\text{Succ}_{cma, cida}^{\text{ema, cida}}$ is negligible for any polynomially bounded $A_{I}$.

**Definition 3**: Let $\text{Succ}_{cma, cida}^{\text{ema, cida}}$ be the success probability of a Super Type II adaptively chosen message and chosen identity adversary $A_{II}$, a certificateless signature scheme is secure against a Super Type II adversary $A_{II}$ if $\text{Succ}_{A_{II}, \text{super}}^{\text{ema, cida}}$ is negligible for any polynomially bounded $A_{II}$.

**Remark**: Due to page limitation, we refer interested readers to [5] for the formal game-based models of above definitions.

**Definition 4**: A certificateless signature scheme is existentially unforgeable against chosen message and chosen identity attacks if it satisfies Def. 2 and Def. 3.

### C. Complexity Assumption

The security of our certificateless signature scheme can be reduced to the hardness of discrete logarithm problem. Let $p, q$ be two primes and $q | (p - 1)$. Let $G$ be a subgroup of $Z_q^*$ with prime order $q$ and generator $g$. The discrete logarithm problem is defined as follows.

**Definition 5**: Given an random element $\beta \in G$, find $\alpha \in Z_q^*$ such that $g^\alpha = \beta \pmod{p}$.

### III. The Proposed Scheme

In this section, we describe our certificateless signature scheme without pairings. It consists of the following algorithms:

- **Setup**: This algorithm runs as follows:
  1) Given a security parameter $k \in \mathbb{N}$, this algorithm first chooses two primes $p, q$, where $p, q > 2^k$ and $q | (p - 1)$. It then chooses an element $g \in Z_q^*$ with order $q$. The subgroup generated by $g$ is denoted as $G$;
  2) The master secret key $x$ is randomly chosen from $Z_q^*$, and the master public key is calculated as $y = g^x \pmod{p}$;
  3) Chooses three distinct cryptographic hash functions $H_1 : \{0, 1\}^* \times Z_q^* \rightarrow Z_q^*$, $H_2 : \{0, 1\}^* \times Z_q^* \times Z_q^* \rightarrow Z_q^*$, $H_3 : \{0, 1\}^* \times \{0, 1\}^* \times (Z_q^*)^4 \times Z_q^* \times Z_q^* \rightarrow Z_q^*$.

  The system parameters $params = (p, q, G, y, H_1, H_2, H_3)$.

- **Partial-Key-Extract**: Given the user’s identity ID $\in \{0, 1\}^*$ as input, this algorithm works as follows:
  1) Picks $s_0, s_1 \in Z_q^*$ at random, and calculates $p_0 = g^{s_0} \pmod{p}$ and $p_1 = g^{s_1} \pmod{p}$;
  2) Calculates $d_0 = s_0 + x \cdot H_1(ID, p_0) \pmod{q}$ and $d_1 = s_1 + x \cdot H_2(ID, p_0, p_1) \pmod{q}$;
  3) The partial private key $D_{ID} = d_0$, the partial public key $PK_{ID} = (p_0, p_1, d_1)$.

**Remark**: Algorithms Setup and Partial-Key-Extract are executed by KGC. Once the partial private and public keys are given to the user via secure channel, the user first checks if $g^{d_0} = p_0 \cdot y^{H_1(ID, p_0)} \pmod{p}$ and $g^{d_1} = p_1 \cdot y^{H_2(ID, p_0, p_1)} \pmod{p}$. If both equations hold, the user continues to run the following algorithms.

- **Set-Secret-Value**: This algorithm picks $z \in Z_q^*$ at random, and sets $s_{ID} = z$ as the user’s secret value.

- **Set-Private-Key**: Given the user’s partial private key $D_{ID}$ and the secret value $s_{ID}$, the full private key $SK_{ID} = (D_{ID}, s_{ID}) = (d_0, z)$.

- **Set-Public-Key**: Given the secret value $s_{ID}$ and the partial public key $PK_{ID} = (p_0, p_1, d_1)$, this algorithm
calculates $\mu = g^r \pmod p$. The user’s public key $PK_{ID} = (PK_{ID}, \mu) = (p_0, p_1, d_1, \mu)$.

- **Sign**: To sign a message $m \in \{0,1\}^*$, the signer
  1. Randomly selects $r, r' \in \mathbb{Z}_q^*$, and calculates $c = g^r \pmod p$ and $c' = g^{r'} \pmod p$;
  2. Sets $u = H_3(m, ID, c, c', PK_{ID})$;
  3. Calculates $v = r - uz \pmod q$ and $w = r' - ud_0 \pmod q$.

The signature on the message $m$ is $\sigma = (u, v, w)$.

- **Verify**: Given params, the signer’s identity $ID$, $PK_{ID} = (p_0, p_1, d_1, \mu)$, a message $m$ and the signature $\sigma = (u, v, w)$, the verifier checks if
  \[
  g^{d_1} = p_1 y^{H_2(ID, p_0, p_1)} \pmod p
  
  u = H_2(m, ID, g^w, g^{w(p_0)^{H_1(ID, p_0)}}, PK_{ID}).
  \]

If both equations are correct, this algorithm outputs "valid". Otherwise, it outputs "invalid".

### IV. Security Analysis

In this section, we will show that the proposed scheme is secure against Super Type I adversary and Super Type II adversary defined in Section II-B. This is ensured by the following two theorems.

**Theorem 1**: Our certificateless signature scheme is secure against Super Type I adversary in the random oracle model, assuming that the discrete logarithm problem is intractable on $\mathbb{Z}_p$.

**Theorem 2**: Our certificateless signature scheme is secure against Super Type II adversary in the random oracle model, assuming that the discrete logarithm problem is intractable on $\mathbb{Z}_p$.

The proofs of above theorems follow the same idea. We shall prove that if there is a Super Type I or Type II adaptively chosen message and chosen identity adversary which can break our certificateless signature scheme with non-negligible probability, then there exists another algorithm $B$ which can solve the discrete logarithm problem with non-negligible success probability as well. Below we give the details of the proof of Theorem 1. Due to page limitation, we omit the proof of Theorem 2, which employs the similar technique to the proof of Theorem 1.

**Proof of Theorem 1**: Let $A_I$ be a Super Type I adversary against our certificateless signature scheme. We want to build an algorithm $B$ that uses $A_I$ as a black-box to solve the discrete logarithm problem. At the beginning, $B$ is given two primes $p, q$ and a discrete logarithm problem instance $(g, \beta = g^\alpha \pmod p)$. The goal of algorithm $B$ is to find $\alpha$ such that $\beta = g^\alpha \pmod p$. We show that by acting as $A_I$’s challenger, $B$ can use $A_I$ to find $\alpha$.

$B$ initializes $A_I$ with the master public key $y = g^x$, where $x$ is the master secret key and $B$ keeps it secret. $B$ then gives system parameters $params = (p, q, g, y, H_1, H_2, H_3)$ to $A_I$. Note that $H_1$ and $H_2$ are real hash functions, but $H_3$ is simulated by $B$ as the random oracle.

At any time, $A_I$ is allowed to access the following oracles in polynomial time. $B$ responds to such queries as follows.

- **Create-User Request**: Suppose $A_I$ makes at most $(q_{CU})$ queries to the Create-User Request oracle. At beginning, $B$ chooses $t \in [1, q_{CU}]$ randomly. For the $i^{th}$ Create-User Request query $ID_i$
  1. If $i \neq t$, $B$ picks $s_0, s_1, z_i \in \mathbb{Z}_q^*$ at random and computes $p_0 = g^{s_0} \pmod p$, $p_1 = g^{s_1} \pmod p$, $(d_0)_{ID_i} = s_0 + x_0 z_i$ and $(d_1)_{ID_i} = s_1 + x_1 z_i$, where $e_0 = H_1(ID_i, (p_0)_{ID_i})$, $e_1 = H_2(ID_i, (p_0)_{ID_i}, (p_1)_{ID_i})$. In this case, the corresponding partial public key of $ID_i$ is $P_{ID_i} = ((p_0)_{ID_i}, (p_1)_{ID_i}, (d_1)_{ID_i})$, $\mu_{ID_i} = g^{z_i} \pmod p$, the partial private key of $ID_i$ is $D_{ID_i} = (d_0)_{ID_i}$, and the secret value $s_{ID_i} = z_i$.
  2. Otherwise, $i = t$ and let $ID_i = ID^*$. $B$ sets $p_0 = \beta = g^\alpha$ and $d_0 = 1$, which means $B$ cannot compute the partial private key of $ID^*$. $B$ then picks $s_1, z_i \in \mathbb{Z}_q^*$ at random and calculates $(p_1)_{ID^*} = g^{s_1} \pmod p$, $(d_1)_{ID^*} = s_1 + x_1 z_i$, where $e_{1i} = H_2(ID^*, (p_0)_{ID^*}, (p_1)_{ID^*})$. In this case, the corresponding partial public key of $ID^*$ is $P_{ID^*} = ((p_0)_{ID^*}, (p_1)_{ID^*}, (d_1)_{ID^*})$, $\mu_{ID^*} = g^{z_i} \pmod p$, and the secret value $s_{ID^*} = z_i$.

In either cases, $B$ adds
\[
(ID_i, D_{ID_i} = (d_0)_{ID_i}, s_{ID_i} = z_i, \quad PK_{ID_i} = ((p_0)_{ID_i}, (p_1)_{ID_i}, (d_1)_{ID_i}, (\mu)_{ID_i}))
\]
on list $L$ and returns $PK_{ID_i}$ to $A_I$.

- **Partial-Private-Key Extraction**: When $B$ receives a partial private key query for a created user $ID_i$,
  1. If $ID_i = ID^*$, $B$ returns “failure” and aborts the simulation.
  2. Otherwise, $ID_i \neq ID^*$. $B$ finds $(d_0)_{ID_i}$ on $L$ and returns $(d_0)_{ID_i}$ as the answer.

- **Secret-Value-Extraction**: For the secret value extraction query on a created user $ID_i$, $B$ finds $s_{ID_i}$ on list $L$ and returns it to $A_I$ as the answer.

- **Public-Key-Replacement Request**: When $A_I$ makes a public key replacement query on $\{ID_i, PK'_{ID_i} = ((p_0'), (p_1'), (d_1'), (\mu')_{ID_i})\}$, $B$ first checks whether
\[
g^{(d_1')_{ID_i}} = (p_1')^{H_2(ID_i)} \cdot y^{H_2(ID_i, p_0')}.\]

If the above equals, $B$ returns “failure” and aborts the simulation. Otherwise, $B$ replaces the original public key $PK_{ID_i}$ with $(PK'_{ID_i})$. Then $B$ will update the list $L$ and rewrites the corresponding information as $(ID_i, D_{ID_i}, s_{ID_i}, (PK'_{ID_i}))$. Note that the secret value and the partial private key corresponding to the new public key are not required.

- **H₃ Oracle Queries**: When $A_I$ makes a query to oracle $H_3$, $B$ first checks the list $L_{H_3}$ to see if there is an entry for the same query. If $(m, ID, c, c', PK_{ID_i}, u)$ appears on $L_{H_3}$, then the same answer $u$ will be given to $A_I$. 376
Otherwise, a new random value $u$ from $\mathbb{Z}_q^*$ will be given as the answer to $A_1$. $B$ then adds it to the list $L_{H_3}$.

- **Super-Sign Queries**: Suppose that $A_1$ makes a signing query on $(ID_1, m)$
  - If $ID_1 \neq ID^*$ and the public key of $ID_1$ has not been replaced, $B$ first finds the corresponding private key $SK_{1ID_1} = ((d'_1)_{ID_1}, s_{1ID_1}, z) = z$ on $L$. Then $B$ uses the private key $SK_{1ID_1}$ to sign the message $B$.
  - Otherwise, though $B$ does not know the private key $SK_{1ID}$, $B$ chooses $(u, v, w) \in (\mathbb{Z}_q^*)^3$ at random and sets $u = H_3(m, ID_1, g^w(\mu)^n, g^w(p_0 g^{H_1(ID_1, p_0)})^v, PK_{1ID})$. If the collision happens, $B$ will rechoose $(u, v, w)$ until there is no collision happens on $L_{H_3}$.

  In either case, $B$ outputs $\sigma = (u, v, w)$ as $ID_1$'s signature on $m$. As the random oracle $H_3$ is controlled by $B$, each signature $\sigma = (u, v, w)$ will pass the verification. The above simulated signature is identically distributed as the one in the real attack. By doing this, $B$ performs a perfect simulation.

  Eventually, $A_1$ outputs a valid signature $(ID, m, \sigma = (u, v, w))$.

- **If $ID = ID^*$ or $\sigma$ is not a valid signature, $B$ returns "failure".
- Otherwise, $B$ can solve the discrete logarithm problem by applying the forking technique. According to the forking lemma [7], if $A_1$ is a sufficient efficient forger in the above interactions, $B$ can obtain two valid signatures $\sigma = (u, v, w)$ and $\sigma' = (u', v', w')$ ($u \neq u'$) that satisfies

$$g^w(p_0 g^{H_1(ID_1, p_0)})^u = g^{w'}(p_0 g^{H_1(ID_1, p_0)})^{u'}$$

$B$ can calculate $\alpha = log_q \beta = log_q(p_0) = (w' - w)/(u - u') - x H_1(ID, \beta)$

**Probability of Success**: It remains to compute the probability that $B$ solves the given instance of the discrete logarithm problem. $B$ succeeds if:

1. $A_1$: $B$ does not abort during the simulation;
2. $A_2$: $(ID, m, \sigma = (u, v, w))$ can pass the verification under the current public key $PK_{1ID}$;
3. $A_3$: In the forgery $(ID, m, \sigma = (u, v, w)), ID = ID^*$.

  This happens with probability $1/q_{CU}$.

$B$ does not abort during the simulation if and only if the following events happen:

1. $A_{11}$: $A_1$ does not make Partial-Private-Key Extraction request of $ID^*$. Suppose that $A_1$ makes at most $q_{PPK}$ queries to the oracle Partial-Private-Key Extraction, this happens with probability $(1 - \frac{1}{q_{CU}})^{q_{PPK}}$.
2. $A_{12}$: $A_1$ does not make a Public-Key-Replacement that satisfies

$$g^{d'_1(ID_1, i)} = (p'_1)^{H_2(ID, p_0, g'_1)}$$

According to forking lemma in [7], if $A_1$ finds another $(p'_0, p'_0, d'_1, u)$ that satisfies $g^{d'_1(ID_1, i)} = (p'_1)^{1ID}, y^{H_2(ID, p_0, g'_1)}$, then the discrete logarithm problem can be solved with probability $\varepsilon \geq \frac{Q}{q}$, in polynomially bounded time, where $Q$ is the number of queries that $A_1$ can ask to the random oracle $H_2$. This will not happen as discrete logarithm is assumed to be hard on $\mathbb{Z}_q^*$.

Therefore, the probability that $B$ can solve the discrete logarithm problem is

$$Adv^{DL}_B \geq 1 - \frac{1}{q_{CU}}(1 - \frac{1}{q_{CU}})^{q_{PPK}}$$

V. Conclusion

In this paper, we present the first concrete certificateless signature scheme without pairings. Our construction is motivated by certificateless encryption schemes without pairings proposed in [2], [12]. The new scheme is provably secure (in the random oracle model) against Super Type I and Super Type II adversaries defined in [5], assuming that the discrete logarithm problem is intractable. The proposed scheme is more computationally efficient than other certificateless signature schemes from bilinear mappings.

**Acknowledgment**

The authors would like to express their gratitude to Qiong Huang, Yinxia Sun and the anonymous reviewers for their valuable suggestions and comments.

This paper is supported by the National Natural Science Foundation of China (NO. 60673081).

**References**


