An Efficient Identity-based Signature Scheme and Its Applications

Shi Cui[1], Pu Duan[1], Choong Wah Chan[1], and Xiangguo Cheng[2]
(Corresponding author: Shi Cui)

Information Security Center, Nanyang Technological University[1]
Nanyang Avenue, Singapore 639798 (Email: cuishi@pmail.ntu.edu.sg)
Institute for Infocomm Research[2]
21 Heng Mui Keng Terrace, Singapore 119613

(Received Nov. 28, 2005; revised and accepted Dec. 31, 2005 & Jan. 10, 2006)

Abstract

Mapping messages or user's identity into a point on elliptic curves is required in many pairing-based cryptographic schemes. In most of these pairing-based schemes, this requirement is realized by a special hash function called *MapToPoint* function. However, the efficiency of the *MapToPoint* function is much lower than the general hash functions. In this paper, we propose a new identity-based signature (IBS) scheme without *MapToPoint* function, which speeds up extracting the secret key and verifying the signatures. The security of the proposed scheme depends on a complex assumption similar to k-CAA. Another benefit of the proposed scheme is that it supports batch verifications such that multiple signatures of distinct messages for distinct users are verified simultaneously. The results show that batch verifications on the proposed IBS scheme is much faster than other IBS schemes. Furthermore, the proposed scheme is used to construct an efficient chameleon signature scheme by cooperating with an identity-based chameleon hash function.

Keywords: ID-based signature, ID-based chameleon signature, batch verification

1 Introduction

The idea of identity-based public key cryptography (ID-PKC) [26] has been proposed for almost twenty years. Although ID-PKC has the ability to simplify the key management in comparison of the traditional public key cryptography (PKC) [21], they were rarely discussed in the real applications for lack of efficient algorithms.

Recently, Boneh and Franklin [6] constructed an efficient identity-based encryption (IBE) scheme by bilinear pairings. Since then, the research on ID-PKC has made great progress. Few variances of the scheme were published, such as identity-based encryption (IBE) schemes [9, 14], identity-based key agreement schemes [10, 28], identity-based signature (IBS) scheme [11, 13, 15, 20, 24, 29, 30]. In particular, IBS has been discussed in the application of securing IPv6 neighbor and router discovery [1]. However, improving the efficiency of IBS scheme is still a interesting research topic.

This paper fist introduces a faster IBS scheme than the existing IBS schemes [11, 15, 20, 24, 29, 30]. In the existing IBS schemes above, a special hash function called *MapToPoint* function [7], which is used to map an identity information (e.g. user name, IP address) into a point on elliptic curve is necessary. This special function is probabilistic and time consuming. Recently, Zhang et al. [32] modified the BLS signature [7] to obtain a fast short signature scheme (ZSS scheme) without the *MapToPoint* function. Motivated by their method, we propose a new IBS scheme without *MapToPoint* function in the random oracle model, which offers better performance than other IBS schemes from pairings. To prove the security of the new IBS scheme, a new complex assumption similar to k-CAA is introduced. Furthermore, a method called batch verifications [4, 30] is discussed for the proposed IBS scheme. By this method, multiple signatures generated by the proposed IBS scheme are verified simultaneously such that the time for the verifications is significantly reduced. Batch verification is classified into three types: Type 1, Type 2 and Type 3. Until now, only one IBS scheme [30] has the ability to support batch verification of Type 3. Fortunately, the proposed IBS scheme also supports the batch verification of Type 3. We will show how batch verification of the new scheme is implemented and provides better performance than [30]. We also described how to construct an efficient identity-based chameleon signature scheme [2, 10] based on the proposed IBS scheme by collaborating with an identity-based chameleon hash function described in [31].

The rest of this paper is organized as follows: Section 2 introduces some basic knowledge of bilinear pairings and the security notion for IBS scheme. Section 3 first presents a new complex assumption, then a new IBS scheme and its security analysis are given. Section 4 describes the speed up of verifications when receiving many signatures generated by the proposed scheme. Section 5 introduces an efficient chameleon signature scheme based on the proposed IBS scheme. The comparison of the performance with other IBS schemes is shown in Section 6. Finally, the conclusion is drawn in Section 7.

We note that in the final stage in the preparation of the paper, Barreto, Libert, McCullagh and Quisquater also independently proposed a similar IBS scheme [3] where a different but excellent security proof is given.

2 Preliminaries

Before describing the new proposed IBS scheme, we first introduce some preliminary knowledge in this section.

2.1 Bilinear Pairing and k-CAA

Suppose G_1 and G_2 are an additive group and a multiplicative group, respectively. They are two cyclic groups of the prime order l. Let P and Q be two distinct generators of G_1. The discrete logarithm problem (DLP) is hard in both G_1 and G_2. Our scheme requires a bilinear pairing, $\hat{e} : G_1 \times G_1 \rightarrow G_2$, which has the following properties:

1) Bilinear: $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$ for all $P, Q \in G_1$ and $a, b \in Z_l^*$.

2) Non-degenerate: there is $\hat{e}(P, P) \neq 1$ for $P \neq \mathcal{O}$.

3) Computable: there exists an efficient algorithm to compute $\hat{e}(P, Q)$ for all $P, Q \in G_1$.

As shown in [5], the modified Tate pairing on a supersingular elliptic curve is such a bilinear pairing.

ZSS scheme [32] depends on a complex assumption: there is no polynomial time algorithm for the Collusion of Attack Algorithm with k Traitors (k-CAA) [19]. The definition of k-CAA is as following:

Definition 1. *For a known $k \in Z$ and an unknown $x \in Z_l^*$, k-CAA is an algorithm which can compute $Q = \frac{1}{x+g}P$ from given (g_1, g_2, $\cdots$, $g_k \in Z_l^*$, $P \in G_1$, xP, $\frac{1}{x+g_1}P$, $\frac{1}{x+g_2}P$, $\cdots$, $\frac{1}{x+g_k}P$), where $g \in Z_l^*$ and not any of $\{g_1, g_2, \cdots, g_k\}$.*

If the tuple (g_1, g_2, $\cdots$, g_k, P, xP, $\frac{1}{x+g_1}P$, $\frac{1}{x+g_2}P$, $\cdots$, $\frac{1}{x+g_k}P$) is given, an algorithm can output $Q = \frac{1}{x+g}P$ for some $g \notin \{g_1, g_2, \cdots, g_k\}$ in at most time t with the possibility at least ϵ. We say that this (t, ϵ)-algorithm can solve k-CAA. Until now, no polynomial time algorithm solves k-CAA.

2.2 The Security of IBS Scheme

IBS scheme includes four algorithms: ***Setup***, ***Extract***, ***Sign*** and ***Verify***. They are used to generate the system parameters, extract the secret key associated the user's identity, sign the message by the secret key and verify the signatures under the public key and the user's identity. In the random oracle model, we say an IBS scheme is existential unforgeable under an adaptive chosen message and identity attack [11, 18] if no polynomial time algorithm $\mathcal{F}$ has non-negligible probability against a challenger $\mathcal{C}$ in the following game:

- **Setup:** The challenger $\mathcal{C}$ runs ***Setup*** to generate the public key and the master key. The public key is sent to the adversary $\mathcal{F}$.
- **Query:** $\mathcal{F}$ makes the following queries:
  1) *Key Extract query:* Given user's identities id_i, $\mathcal{C}$ outputs the corresponding private keys by running ***Extract***.
  2) *Message hash query:* $\mathcal{C}$ computes the hash value of the message m_j and sends them to $\mathcal{F}$.
  3) *Sign query:* Given (id_i, m_j), $\mathcal{C}$ outputs signature σ by running ***Sign*** and sends them to $\mathcal{F}$.
- **Output:** $\mathcal{F}$ outputs (id, m, σ) and wins the game if
  1) (id, m, σ) is a valid signature;
  2) id is not any of id_i and (id, m) is not any of (id_i, m_j) in the step of *Sign Query*.

  Otherwise, $\mathcal{F}$ stops and outputs failure.

Let ϵ denote the probability that $\mathcal{F}$ wins the above game, we have the following definition:

Definition 2. *In the random oracle model, an algorithm $\mathcal{F}$ can $(t, q_H, q_E, q_S, \epsilon)$-breaks an IBS scheme if $\mathcal{F}$ outputs a forgery with probability at least ϵ by running in time at most t, making at most q_H queries to the hash oracle, q_E extract queries, q_S signature queries. An IBS signature scheme is $(t, q_H, q_E, q_S, \epsilon)$-existential unforgeable under an adaptive chosen message and identity attack if no algorithm $(t, q_H, q_E, q_S, \epsilon)$-breaks it.*

2.3 Batch Verifications and its Security

The goal of batch verifications is to verify the multiple signatures simultaneously such that the time for verifications is reduced. Its definition is:

Definition 3. *Given multiple signatures σ_1, σ_2, $\cdots$, σ_n on the messages m_1, m_2, $\cdots$, m_n and the corresponding identities id_1, id_2, $\cdots$, id_n, a verifier checks the validity of some of or all the signatures at once.*

There are three types of batch verifications [30]:

- **Type 1:** Multiple signers sign a single message to obtain multiple signatures.

- **Type 2:** A single signer signs multiple messages to obtain multiple signatures.
- **Type 3:** Multiple signers sign multiple messages to obtain multiple signatures. Note that all the messages are distinct, so are the signers.

Yoon et al. [30] formalized the notion of the attack model of batch verifications of Type 1, 2 and 3 on the general IBS scheme. We say that $\mathcal{F}$ is a λ-batch forger of Type 1, 2 and 3 if it wins the following game:

- **Setup:** $\mathcal{F}$ is given public parameters.
- **Queries:** $\mathcal{F}$ accesses the hash, extract and sign oracle by his choices and obtains the hash values of his queries, the secret keys of his chosen identities and the signatures of his chosen identities and messages.
- **Outputs:** Finally, $\mathcal{F}$ outputs an integer n whose value is not larger than λ, $id_1, id_2, \cdots, id_n$ and messages $m_1, m_2, \cdots, m_n$ and the corresponding signatures $\sigma_1, \sigma_2, \cdots, \sigma_n$ of Type 1, 2 and 3. Note that id_n must not be queried by the extract oracle, (id_n, m_n) must not be queried by the sign oracle. $\mathcal{F}$ wins the game if $\mathcal{F}$'s outputs pass the batch verifications successfully.

In the game above, note that $\mathcal{F}$ is given the power to access all the users' private keys except id_n and access the sign oracle on all the messages except m_n. From the description above, the following definition is given:

Definition 4. *In the random oracle model, a λ-batch forger $\mathcal{F}$ $(t, q_H, q_E, q_S, \lambda, \epsilon)$-breaks the batch verifications on some IBS scheme by the adaptive chosen message and identity attack if $\mathcal{F}$ runs in time at most t, makes at most q_H queries to the hash oracle, q_E extract queries and q_S signature queries with the probability at least ϵ to generate at most λ signatures which pass successfully the batch verifications.*

3 The Proposed Identity-based Signature Scheme

In this section, a new complexity assumption is first introduced. We then describe the new IBS scheme and its security analysis.

3.1 Generalized k-CAA

Before introducing the new IBS scheme, we first propose a new complex assumption, here called Generalized k-CAA:

Definition 5. *For a known $k \in Z$ and an unknown $x \in Z_l^*$, k is the product of two integers m and n, Generalized k-CAA is an algorithm which computes $\frac{f}{x+g}P$ from a given tuple ($f_1, f_2, \cdots, f_m, g_1, g_2, \cdots, g_n \in Z_l^*$, $P \in G_1$, xP, $\frac{f_1}{x+g_1}P$, $\frac{f_2}{x+g_1}P$, $\cdots$, $\frac{f_m}{x+g_1}P$, $\frac{f_1}{x+g_2}P$, $\cdots$, $\frac{f_m}{x+g_{n-1}}P$, $\frac{f_1}{x+g_n}P$, $\cdots$, $\frac{f_m}{x+g_n}P$), where $f, g \in Z_l^*$, $f \notin \{f_1, f_2, \cdots, f_m\}$ and $g \notin \{g_1, g_2, \cdots, g_n\}$.*

If the tuple ($f_1, f_2, \cdots, f_m, g_1, g_2, \cdots, g_n \in Z_l^*$, $P \in G_1$, xP, $\frac{f_1}{x+g_1}P$, $\frac{f_2}{x+g_1}P$, $\cdots$, $\frac{f_m}{x+g_1}P$, $\frac{f_1}{x+g_2}P$, $\cdots$, $\frac{f_m}{x+g_{n-1}}P$, $\frac{f_1}{x+g_n}P$, $\cdots$, $\frac{f_m}{x+g_n}P$) is given, an algorithm outputs $Q = \frac{f}{x+g}P$ for $f \notin \{f_1, f_2, \cdots, f_m\}$ and $g \notin \{g_1, g_2, \cdots, g_n\}$ in at most time t with the possibility at least ϵ. We say that this (t, ϵ)-algorithm can solve the Generalized k-CAA. Let $f_i = 1$, the Generalized k-CAA is transformed into n-CAA. Thus, k-CAA can be seen as a special case of the Generalized k-CAA. From the description above, the following lemma is yielded:

Lemma 1. *There is no polynomial time algorithm for solving the Generalized k-CAA.*

Proof. Suppose that there is a polynomial time algorithm can solve the Generalized k-CAA. From the description above, this algorithm must solve n-CAA, too. We know that there is no polynomial time algorithm for solving n-CAA, therefore, the supposal is not correct. □

3.2 The Proposed IBS Scheme

In the existing IBS schemes from bilinear pairings [11, 15, 20, 29, 30], extracting the secret key from the master key and the user's identity requires a special hash function called *MapToPoint* function [7] which maps the user's identity id (where $id \in Z_l^*$) into an element of G_1. Recently, in the papers of Mitsunari et al. [19] and Zhang et al. [31], another method for generating the secret key S_{id} from the master key $x \in Z_l^*$ and the user's identity id: $S_{id} = \frac{1}{x+id}P$ for $P \in G_1$(or $S_{id} = \frac{1}{x+H(id)}P$, where H is a general hash function). Using this method of generating the secret key, a new IBS scheme without *MapToPoint* function is constructed. The scheme is described as follows:

- ***Setup:*** the trust authority (**TA**) chooses randomly $P \in G_1$ and $x \in Z_l^*$, compute $P_{pub} = xP$ and precompute $\omega = \hat{e}(P, P)$. x is the master key. The public key is (P, P_{pub}, ω, H), where $H : \{0,1\}^* \times G_2^* \to Z_l^*$ is a hash function.
- ***Extract:*** For a given identity $id \in Z_l^*$, **TA** computes the secret key $S_{id} = \frac{1}{x+id}P$. Note if $x + id \equiv 0 \pmod{l}$, then abort x and return ***Setup*** to choose another x.
- ***Sign:*** Given the secret key S_{id} and the message $m \in \{0,1\}^*$, the signer chooses a random element s from Z_l^* and computes $r = \omega^s$, $u = H(m, r)$, $v = (u + s)S_{id}$. The signature pair (r, v) is sent to the verifier.
- ***Verify:*** Given the public key (P, P_{pub}, ω, H), a message m, a user's identity id and a signature pair (r, v), the verifier computes $u = H(m, r)$, and accepts the signature if $\omega^u r = \hat{e}(P_{pub} + id \cdot P, v)$.

Note that ***Extract*** is only done once for every identity. The procedure of the verification is deduced as follows:

$$\begin{aligned}\hat{e}(P_{pub} + id \cdot P, v) &= \hat{e}((x + id)P, S_{id})^{(u+s)} \\ &= \omega^{u+s} \\ &= \omega^u r.\end{aligned}$$

3.3 Security Analysis

In the existing IBS schemes [11, 15, 29, 30], the forking lemma [22, 23] is necessary for proving the security of the schemes. But the the use of the forking lemma cannot yield tight security reductions [18]. Recently, some signature schemes [8, 32, 33] have been proved secure under the adaptive chosen message attack but the forking lemma is not used in their proof. In this section, we follows their method to prove the security of the proposed IBS scheme under the adaptive chosen message attack.

To prove that the security of the proposed scheme depends on the Generalized k-CAA, the following theorem is given:

Theorem 1. *In the random oracle model, if an algorithm $\mathcal{F}$ $(t, q_H, q_E, q_S, \epsilon)$-breaks the proposed scheme under the adaptive chosen message and identity attack, then there is another $(t'$, $\epsilon')$ -algorithm $\mathcal{C}$ which can solve the Generalized k-CAA, where $t' = t$, $q_S \leq q_H$, $k = q_E \cdot q_S$ and $\epsilon' = \frac{(l-q_S)(q_S)^{q_E \cdot q_S}}{l(q_H)^{q_E \cdot q_S}} \cdot \epsilon$.*

Proof. Suppose that an algorithm $\mathcal{F}$ $(t, q_H, q_E, q_S, \epsilon)$-breaks the proposed scheme by the adaptive chosen message and identity attack. We expect to construct an algorithm $\mathcal{C}$ to solve the Generalized k-CAA from $\mathcal{F}$. Namely, given a tuple $(f_1, f_2, \cdots, f_m, g_1, g_2, \cdots, g_n \in Z_l^*, P \in G_1, xP, \frac{f_1}{x+g_1}P, \frac{f_2}{x+g_1}P, \cdots, \frac{f_m}{x+g_1}P, \frac{f_1}{x+g_2}P, \cdots, \frac{f_m}{x+g_{n-1}}P, \frac{f_1}{x+g_n}P, \cdots, \frac{f_m}{x+g_n}P)$, $\mathcal{C}$ has an ability of outputting $\frac{f}{x+g}P$ for $f \notin \{f_1, f_2, \cdots, f_m\}$, $g \notin \{g_1, g_2, \cdots, g_n\}$. In the following simulation, $\mathcal{F}$ and $\mathcal{C}$ play the role of the adversary and the challenger, respectively. $\mathcal{F}$ will interact with $\mathcal{C}$ as follows:

- **Setup:** $\mathcal{C}$ runs ***Setup*** to obtain the public key (P, Q, ω, H) where $Q = xP$. $x \in Z_l^*$ is the master key. The public key is sent to $\mathcal{F}$.
- **Query:** $\mathcal{F}$ issues the following queries for the identities $(id_1, id_2, \cdots, id_{q_E})$ and the messages $(m_1, m_2, \cdots, m_{q_S})$:
  1) *Key Extract Query:* For any given identity id_i ($1 \leq i \leq q_E$), $\mathcal{C}$ computes its corresponding secret key $S_{id_i} = \frac{1}{(x+id_i)}P$, then send it to $\mathcal{C}_0$.
  2) *Message Hash Query:* For any given message m_j $(1 \leq j \leq q_H)$, $\mathcal{C}$ constructs a L_1-list of tuple $< m_j, r_j, u_j, s_j >$ for responding $\mathcal{F}$'s queries. When $\mathcal{F}$ sends a hash query for the message m_j, $\mathcal{C}$ picks two random elements s_j and u_j from Z_l^* such that $s_i + u_i \neq s_j + u_j$ when $i \neq j$, then computes $r_j = \omega^{s_j}$. Let $u_j = H(m_j, r_j)$. u_j is sent to $\mathcal{F}$ as the response of the hash query on the message m_j. Simultaneously, $\mathcal{C}$ constructs another L_2-list $\{h_1, h_2, \cdots, h_{q_H}\}$ where $h_j = u_j + s_j$.
  3) *Sign Query:* For any given identity-message pair (id_i, m_j) where $1 \leq i \leq q_E$ and $1 \leq j \leq q_H$, $\mathcal{C}$ first runs the hash query algorithm to check whether m_j appears in the L_1-list. If it is not, $\mathcal{C}$ stops the simulation and reports failure. Otherwise, $\mathcal{C}$ obtains the corresponding r_j, u_j and s_j from L_1 and computes
  $$v_{ij} = (u_j + s_j)S_{id_i} = \frac{u_j + s_j}{x + id_i}P.$$
  $\mathcal{C}$ finds h_k from L_2-list such that $h_k = u_j + s_j$ (where $1 \leq j \leq q_S$, $1 \leq k \leq q_H$, $q_S \leq q_H$), then the pair $(r_j, \frac{h_k}{x+id_i}P)$ is viewed as the signature on the message m_j for the user id_i from $\mathcal{F}$'s point of view. $\mathcal{C}$ return it to $\mathcal{F}$ as the response of the sign oracle.
- **Output:** Finally, $\mathcal{F}$ outputs a pair (r^*, v^*) on the message m^* for the user id^*, and accepts it if the follows are satisfied:
  1) $id^* \notin \{id_1, id_2, \cdots, id_{q_E}\}$ and $m^* \notin \{m_1, m_2, \cdots, m_{q_H}\}$;
  2) (id^*, m^*, r^*, v^*) can successfully pass the check of ***verify*** under the public key.

Suppose $r^* = \omega^{s^*}$ and $H(m^*, r^*) = u^* \in Z_l^*$ such that $h^* = u^* + s^* \notin \{h_1, h_2, \cdots, h_{q_H}\}$, where s^* and u^* are two random elements in Z_l^*. Since $\mathcal{F}$'s output (id^*, m^*, r^*, v^*) is a valid signature, there is

$$\begin{aligned}\hat{e}(Q + id^* \cdot P, v^*) &= \omega^{u^*} r^* \\ \Rightarrow \quad \hat{e}(P, v^*)^{(x+id^*)} &= \hat{e}(P, P)^{(u^*+s^*)}.\end{aligned}$$

Therefore, $v^* = \frac{u^*+s^*}{x+id^*}P = \frac{h^*}{x+id^*}P$. From $\mathcal{C}$'s point of view, $v^* = \frac{h^*}{x+id^*}P$ is viewed as the solution of the Generalized k-CAA. The reason is as follows: When $m = q_S$ and $n = q_E$, namely $k = q_E \cdot q_S$, $\mathcal{C}$ can compute v^* from the known tuple $(h_1, h_2, \cdots, h_{q_S}, id_1, id_2, \cdots, id_{q_E} \in Z_l^*, P \in G_1, xP, \frac{h_1}{x+id_1}P, \frac{h_2}{x+id_1}P, \cdots, \frac{h_{q_S}}{x+id_1}P, \frac{h_1}{x+id_2}P, \cdots, \frac{h_{q_S}}{x+id_{q_E-1}}P, \frac{h_1}{x+id_{q_E}}P, \cdots, \frac{h_{q_S}}{x+id_{q_E}}P)$ where h_i is from the response of the message hash query on the message m_i, the pair (m_i, id_j) is random by $\mathcal{F}$'s adaptive choices.

Since the hash function behaves as a random oracle, $\mathcal{F}$ is not sure whether $\mathcal{C}$ is a simulator or a real attacker. The running time t' of $\mathcal{C}$ is the same as t of $\mathcal{F}$. In the step of *Sign Query*, $\mathcal{C}$ stops the simulation and report failure only when m_j is not in the L_1. The probability that this event doesn't happen is $\frac{q_S}{q_H}$. For all the q_S sign queries, $\mathcal{C}$'s success probability is $(\frac{q_S}{q_H})^{q_E \cdot q_S}$. Furthermore, the probability of another independent event, $h^* = u^* + s^* \notin \{h_1, h_2, \cdots, h_{q_H}\}$, is (1 - $\frac{q_S}{l}$). Hence, $\mathcal{C}$'s success probability ϵ' is $\frac{(l-q_S)(q_S)^{q_E \cdot q_S}}{l(q_H)^{q_E \cdot q_S}} \cdot \epsilon$.

□

Remark 1. *We can modify the proposed scheme such that the proposed IBS scheme provides shorter signature. The modification is that the signer sends* (h, v) *as the signature. We note that the security of the modification scheme is the same as the original scheme. But the modified scheme is not suitable for the following batch verifications.*

4 Batch Verification

Recently, Yoon et al. [30] used a method called batch verifications to speed up the verification of the signatures generated by their IBS scheme. In fact, it is more precise to call this method *signature screening* [4]. The reason has been described in [4]: This method is not used to determine whether every signature for verification is the correct one of the corresponding message but determine whether the signer has at some point authenticated the messages for verifications. *Signature screening* is a very useful tool in the real applications [30]. Some examples have been shown in [30].

As shown in [30], batch verification of Type 2 has been support by most existing IBS schemes, but only the IBS scheme in [30] supports batch verification of Type 3 until now. Fortunately, the proposed IBS scheme supports both Types 2 and 3 with the better performance. The following shows how to implement batch verifications of Types 2 and 3 on the proposed scheme.

- ***Batch Verification for Type 2:*** Suppose a signer with the identity id generates the signatures (r_1, v_1), (r_2, v_2), $\cdots$, (r_λ, v_λ) on the at most λ distinct messages m_1, m_1, $\cdots$, m_λ. Then the verifier can verify these signatures simultaneously by the following:

$$\begin{aligned} u_i &= H(m_i, r_i) \\ \omega^{\Sigma_{i=1}^{\lambda} u_i} \prod_{i=1}^{\lambda} r_i &= \hat{e}(P_{pub} + id \cdot P, \sum_{i=1}^{\lambda} v_i). \end{aligned}$$

- ***Batch Verification for Type 3:*** Suppose there are at most λ signatures (id_1, m_1, r_1, v_1), (id_2, m_2, r_2, v_2), $\cdots$, $(id_\lambda, m_\lambda, r_\lambda, v_\lambda)$ where all the messages are distinct, so are the identities. Then the verifier can verify these signatures simultaneously by the following:

$$\begin{aligned} u_i &= H(m_i, r_i), \\ \omega^{\Sigma_{i=1}^{\lambda} u_i} \prod_{i=1}^{\lambda} r_i &= \hat{e}(P_{pub}, \sum_{i=1}^{\lambda} v_i) \hat{e}(P, \sum_{i=1}^{\lambda} id_i \cdot v_i). \end{aligned}$$

In the next section, we concentrate on proving the security of batch verification of Type 3 of the proposed scheme. The proof of the security of batch verification of Type 2 is similar.

4.1 The Security of Batch Verifications for Type 3

The security of batch verifications of Type 3 on the proposed IBS scheme depends on the following theorem:

Theorem 2. *In the random oracle model, if a* λ*-batch forger* $\mathcal{F}$ $(t, q_H, q_E, q_S, \lambda, \epsilon)$*-breaks the batch verifications of Type 3 on the proposed scheme under the adaptive chosen message and identity attack, then there is another* (t', ϵ')*-algorithm* $\mathcal{C}$ *which has ability of solving the Generalized* k*-CAA, where* $t' = t$, $q_S \leq q_H$, $k = q_S$ *and* $\frac{(l-q_S)(q_S)^{q_S}}{l(q_H)^{q_S}} \cdot \epsilon$.

Proof. Suppose the algorithm $\mathcal{F}$ is a λ-batch forger that $(t, q_H, q_E, q_S, \lambda, \epsilon)$-breaks the proposed IBS scheme. We wish to construct another algorithm $\mathcal{C}$ to solve the Generalized k-CAA. In the following game, $\mathcal{C}$ plays the role of challenger and interacts with the forger $\mathcal{F}$:

- **Setup:** Algorithm $\mathcal{C}$ runs ***Setup*** and sends $\mathcal{F}$ the public key (P, Q, ω, H), where $Q = xP$ and x is a random element in Z_l^*.
- **Queries:** $\mathcal{F}$ makes the following queries
  1) *Key Extract Query:* Algorithm $\mathcal{F}$ queries the extract oracle by his chosen identities id_i, where $1 \leq i \leq q_E$. $\mathcal{C}$ responds the corresponding private keys $S_{id_i} = \frac{1}{x+id_i} P$.
  2) *Message Hash Query:* $\mathcal{C}$ constructs a H-list of tuple $< m_i, r_i, u_i, s_i >$ $(1 \leq i \leq q_H)$ for responding $\mathcal{F}$'s queries on *the message hash query.* When the adversary $\mathcal{F}$ queries the hash oracle on the message m_i, the H-list is changed as follows: If $\mathcal{F}$ sends a query for message m_i which has appeared in H-list, then $\mathcal{C}$ answers the corresponding (r_i, u_i, s_i) to $\mathcal{F}$. Otherwise, $\mathcal{C}$ picks a random element $s_i \in Z_l^*$ and a random element $u_i \in Z_l^*$, then computes $r_i = w^{s_i}$. Let u_i $= H(m_i, r_i)$ such that $s_i + u_i \neq s_j + u_j$ when $i \neq j$. Each $< m_i, r_i, u_i, s_i >$ is added into the H-list. In addition, $\mathcal{C}$ maintains another set S $= \{h_1, h_2, \cdots, h_{q_H}\}$ where $h_i = u_i + s_i$.
  3) *Sign Query:* For any given identity-message pair $(id_i,\ m_j)$, $\mathcal{C}$ responds $\mathcal{F}$'s queries on the sign oracle as follows: $\mathcal{C}$ scans the H-list to check whether m_j is in the list or not. If it is not, $\mathcal{F}$ stops the simulation and reports failure. Otherwise, $\mathcal{F}$ obtains the corresponding r_j, u_j, s_j. Since $\mathcal{F}$ is λ-batch forger of Type 3 that requires multiple signatures on multiple messages generated by multiple signers, a distinct message must be signed by a distinct user. There is a one-to-one map relationship between the user set U: $\{id_1, id_2, \cdots, id_{q_E}\}$ and the message set M: $\{m_1, m_2, \cdots, m_{q_S}\}$. We might as well think that the signature on the message m_i for the user id_j is discarded if $i \neq j$. Suppose $\mathcal{C}$ computes $\delta_j = u_j + s_j$ such that $\delta_j \in \{h_1,\ h_2,\ \cdots,$

$h_{q_H}\}$ ($q_S \le q_H$), then computes the signature $v_j = \delta_j S_{id_j}$. Otherwise, $\mathcal{C}$ stops the simulation and report failure. Finally, r_j and v_j are sent to $\mathcal{F}$ as the response of the *sign query*.

- **Output:** Eventually, $\mathcal{F}$ stops the simulation and returns the following values: a value n, n identities id_1, $id_2, \cdots, id_n$, n messages $m_1, m_2, \cdots, m_n$ and n signatures $(r_1, v_1), (r_2, v_2), \cdots, (r_n, v_n)$. Notes that id_n and m_n must not be queried by the extract oracle and the sign oracle, respectively. The corresponding H-list is $< m_i, r_i, u_i, s_i >$ where $1 \le i \le (n-1)$. $\mathcal{F}$ wins the game only if the following conditions are satisfied:

  1) $\mathcal{F}$'s outputs pass the batch verifications,
  2) There is a one-to-one map between the user set U and the message set M. The distinct message must be signed for the distinct user.

  Suppose $r_n = \omega^{s_n}$, let $u_n = H(m_n, r_n)$, where s_n and u_n are randomly chosen in Z_l^* such that $\delta_n = u_n + s_n \notin \{h_1, h_2, \cdots, h_{q_H}\}$. Since $\mathcal{F}$'s outputs, $(id_1, m_1, r_1, v_1), (id_2, m_2, r_2, v_2), \cdots, (id_n, m_n, r_n, v_n)$ pass the batch verifications. There is

$$\begin{aligned}&\omega^{\Sigma_{i=1}^{n} u_i}\prod_{i=1}^{n} r_i\\ =\;& \hat{e}(P_{pub}, \sum_{i=1}^{n-1} v_i + v_n)\hat{e}(P, \sum_{i=1}^{n-1} id_i \cdot v_i + id_n \cdot v_n).\end{aligned} \quad (1)$$

  In addition, $(id_1, m_1, r_1, v_1), (id_2, m_2, r_2, v_2), \cdots, (id_{n-1}, m_{n-1}, r_{n-1}, v_{n-1})$ must pass the batch verifications. Therefore, the following formula is correct:

$$\omega^{\Sigma_{i=1}^{n-1} u_i}\prod_{i=1}^{n-1} r_i = \hat{e}(P_{pub}, \sum_{i=1}^{n-1} v_i)\hat{e}(P, \sum_{i=1}^{n-1} id_i \cdot v_i). \quad (2)$$

  Since $\omega = \hat{e}(P, P)$ and $r_i = w^{s_i}$, combine Equations (1) with (2):

$$\hat{e}(P, P) = \hat{e}((x + id_n)P, v_n)^{1/(u_n+s_n)}.$$

  Hence, $v_n = \frac{u_n+s_n}{x+id_n}P = \frac{\delta_n}{s+id_n}P$. Since $\delta_n \notin \{h_1, h_2, \cdots, h_{q_H}\}$ and id_n is not queried by the extract oracle, $\mathcal{C}$ outputs v_n as the solution of the Generalized k-CAA (Actually, v_n is the solution of a special instance of the Generalized k-CAA: given a tuple (f_1, $f_2, \cdots, f_k, g_1, g_2, \cdots, g_k \in Z_l^*$, $P \in G_1$, xP, $\frac{f_1}{x+g_1}P$, $\frac{f_2}{x+g_2}P, \cdots, \frac{f_k}{x+g_k}P$), where $f, g \in Z_l^*$, $f \notin \{f_1, f_2, \cdots, f_k\}$ and $g \notin \{g_1, g_2, \cdots, g_k\}$, compute $\frac{f}{x+g}P$.

$\mathcal{C}$ aborts the simulation only when $\delta_i \notin \{h_1, h_2, \cdots, h_{q_H}\}$. The probability that $\mathcal{F}$'s outputs pass batch verifications is at least q_S/q_H. Thus, for all sign queries, the probability that $\mathcal{C}$'s outputs pass batch verifications is at least $(q_S/q_H)^{q_S}$. The probability of another event, $\delta_n \notin \{h_1, h_2, \cdots, h_{q_H}\}$, is $1 - \frac{q_S}{l}$. The probability that $\mathcal{C}$ successfully outputs the solution of k-CAA is $\frac{(l-q_S)(q_S)^{q_S}}{l(q_H)^{q_S}} \cdot \epsilon$. $\mathcal{C}$'s running time is identical to $\mathcal{F}$'s running time, $t = t'$. □

5 ID-based Chameleon Signature Scheme

The concept of the chameleon signature was first introduced in [17]. Ateniese and Medeiros [2] then designed the identity-based chameleon signature. Such signature provides non-transferability: Any third party cannot accept the signature that has been issued to a designated recipient. It is very similar with undeniable signature [12], but the verifier has the ability to verify the signature without interacting with the signer. On the other hand, the signer also has the ability to deny the validity of the signature by revealing certain values [2]. This is based on a trapdoor one-way hash function: chameleon hash function. Without knowledge of the associated trapdoor, the chameleon hash function is resistant to the computation of pre-images and of collisions. In contrast, with the knowledge of the trapdoor, anyone will compute easily the collisions.

5.1 ID-based Chameleon Hash Scheme from Pairings

Zhang et al [31] introduced two Chameleon hash schemes from bilinear pairings: *Scheme 1* and *Scheme 2*. Based on *Scheme 1*, a Chameleon signature scheme over Cha-Cheon's IBS scheme [11] is given. *Scheme 2* is also used to construct ID-based Chameleon Signature Scheme over Cha-Cheon's IBS scheme [11]. However, **TA** has to generate two different private keys for the same identity. The reason is that extracting the private key associated with the identity of the Chameleon hash scheme is different from that of the signature scheme. *Scheme 2* requires extracting the private key by $S_{id} = \frac{1}{s+H_1(id)}P$ where $H_1(x)$ is a general cryptographic hash function (e.g. SHA hash function), but the signature scheme requires extracting the private key by $S_{id} = sH_0(ID)$, where $s \in Z_l^*$ is the master key, $H_0(x)$ is so called *MapToPoint* function. In the following, we first review *Scheme 2* and make a slight modification by eliminate the general hash function $H_1(x)$ in the extracting secret key such that it is the same as the proposed IBS scheme. In addition, a print error of *Scheme 2* in [31] is corrected.

Setup: **TA** chooses a random member $x \in Z_l^*$ and computes $P_{pub} = xP$. $H_1 : \{0, 1\}^* \mapsto Z_l^*$, is a general hash function. **TA** publish $\{G_1, G_2, \hat{e}, P, P_{pub}\}$ as the public parameters, x is kept as the master key.

Extract: For the given user identity $id \in Z_l^*$, compute the corresponding private key $S_{id} = \frac{1}{x+id}P$. **TA** will choose another x if $x + id \equiv 0 \pmod{l}$.

Hash: For a given message m, choose a random element R from G_1, define the hash as

$$Hash(id, m, R) = \hat{e}(P, P)^{H_1(m)}\hat{e}(idP+P_{pub}, R)^{H_1(m)}.$$

Forge: The Forge algorithm is

$$\begin{aligned} & Forge(id, S_{id}, m, R, m') = R' \\ = \; & H_1(m')^{-1}((H_1(m) - H_1(m'))S_{id} + H_1(m)R). \end{aligned}$$

This forgery is right for the following deduction:

$$\begin{aligned} & Hash(id, m', R') \\ = \; & \hat{e}(P, P)^{H_1(m')}\hat{e}(id{\cdot}P + P_{pub}, R')^{H_1(m')} \\ = \; & \hat{e}(P, H_1(m')P)\hat{e}(id{\cdot}P \\ & +P_{pub}, H_1(m')H_1(m')^{-1}((H_1(m) - H_1(m'))S_{id} \\ & +H_1(m)R)) \\ = \; & \hat{e}(P, H_1(m')P)\hat{e} \\ & (id{\cdot}P + P_{pub}, (H_1(m) - H_1(m'))S_{id} \\ & +H_1(m)R)) \\ = \; & \hat{e}(P, H_1(m')P)\hat{e} \\ & (id \cdot P + P_{pub}, (H_1(m) - H_1(m'))S_{id})\hat{e}(id{\cdot}P \\ & +P_{pub}, H_1(m)R)) \\ = \; & \hat{e}(P, H_1(m')P)\hat{e}(P, (H_1(m) - H_1(m'))P)\hat{e} \\ & (id{\cdot}P + P_{pub}, H_1(m)R)) \\ = \; & \hat{e}(P, P)^{H_1(m)}\hat{e}(id{\cdot}P + P_{pub}, R)^{H_1(m)}. \end{aligned}$$

From the description of *Scheme 2* in [31], the hash is defined as

$$Hash(id, m, R) = \hat{e}(P, P)^{H_1(m)}\hat{e}(H_1(id) + P_{pub}, R)^{H_1(m)},$$

where $H_1(x)$ is a general cryptographic hash function from a string $\{0, 1\}^*$ to Z_l^*. $H_1(id)$ is an element of Z_l^*, but P_{pub} is an element of G_1. The addition between an element of Z_l^* and an element of G_1 is impossible. The correct formula should be as

$$\begin{aligned} & Hash(id, m, R) \\ = \; & \hat{e}(P, P)^{H_1(m)}\hat{e}(H_1(id) \cdot P + P_{pub}, R)^{H_1(m)}. \end{aligned}$$

By the modification above, the deduction of the forgery in [31] is correct. This modification doesn't affect the correctness of Claim 2 in [31]. In this paper, the identity id is redefined an element of Z_l^* instead of a binary string being transferred as an element of Z_l^* by a hash function $H_1(x)$ in *Scheme 2* in [31]. In the modified version, this hash function is omitted because it doesn't influence the security of the Chameleon hash scheme.

5.2 New ID-based Chameleon Signature Scheme

Setup: The trusted authority picks a random x from Z_l^*, and computes $P_{pub} = xP$.

Table 1: Timings of the cryptographic primitives

Primitives	I	M_{G_1}	H_M	P	E	A	M_{G_2}
Timing (ms)	0.03	6.83	3.00	47.40	3.13	0.06	0.03

Extract: Alice is the signer with the public key id_A and private key $S_{id_A} = \frac{1}{x+id_A}P$, Bob is the signer with the public key $S_{id_B} = \frac{1}{x+id_B}P$ and private key S_{id_B}.

Sign: For a given message, Alice picks a random s in Z_l^* and a random element R in G_1, compute $r = \omega^s$, and

$$\begin{aligned} h \; & = \; hash(id_B, m, R) \\ & = \; \omega^{H_1(m)}\hat{e}(id_B \cdot P + P_{pub}, R)^{H_1(m)}. \end{aligned}$$

Then, compute $u = H(h, r)$ and $v = (u+s)S_{id_A}$. The signature (u, v, R) is sent to the verifier.

Verify: The verifier computes $r = \hat{e}(P_{pub} + id_A \cdot P, v)\omega^{-u}$, and accepts the signature if

$$u = H(hash(id_B, m, R), r).$$

Where the function *hash* is the Chameleon hash function. The unforgeability of this chameleon signature scheme still depends on the security of the proposed IBS scheme and the Scheme 2.

6 Performance Comparison

In this section, we first compare our proposed IBS scheme with other IBS schemes [24, 20, 15, 11, 29, 30] in respect to efficiency. We then show how batch verification of Type 3 on our scheme offers better performance than other IBS schemes.

The proposed IBS scheme requires a bilinear pairing with the property $\hat{e}(P, P) \neq 1$. Consider that the cost of the exponentiation on G_2 is very time consuming when the embedding degree is large [16, 25]. Thus, we choose a subgroup of order l in a supersingular elliptic curve $E(\mathbb{F}_p)$ with the embedding degree 2, where l is a 160-bit prime and p is 512-bit prime. Timings for some cryptographic primitives over $\mathbb{F}_p$, G_1 and G_2 are shown in Table 1 where I, M_{G_1}, H_M, P, E, A and M_{G_2} denote the cost of computing an inverse operation over $\mathbb{F}_p$, a scalar multiplication in G_1, the *MapToPoint* function, the pairing, an exponentiation in G_2, a point addition on G_1 and a multiplication on G_2, respectively. All the implementation of these primitives are provided by Miracl [27] on Pentium IV 2.26GHz with 256M RAM. The results in Table 1 indicate that the cost of I, A and M_{G_2} are trivial in comparison with other primitives. Thus, they are usually omitted in the following analysis except mentioning them.

[15] has showed that Hess' scheme provided advantage over the other scheme [11, 20, 24] in term of the efficiency.

Table 2: The comparison of the proposed scheme and other IBS schemes

Scheme	Proposed scheme	Hess [15]	Yi [29]	YCK [30]
Precomputation	$1P$	$1P$	N/A	N/A
Setup	$1M_{G_1}$	$1M_{G_1}$	$1M_{G_1}$	$1M_{G_1}$
Extract	$1I + 1M_{G_1}$	$1H_M + 1M_{G_1}$	$1H_M + 1M_{G_1}$	$1H_M + 1M_{G_1}$
Sign	$1M_{G_1} + 1E$	$1M_{G_1} + 1E$	$3M_{G_1}$	$1H_M + 3M_{G_1}$
Verify	$1M_{G_1} + 1E + 1P$	$1H_M + 2P + 1E$	$1H_M + 1M_{G_1} + 2P$	$1H_M + 1M_{G_1} + 2P$
Signature size	$G_1 \times Z_l^*$	$G_1 \times Z_l^*$	G_1	$G_1 \times G_1$

Hence, only Hess' scheme in these IBS scheme is considered in the Table 2. Besides [11, 15, 20, 24], Yi [29] also proposed an IBS scheme with the shortest signature. Another IBS scheme is also compared in Table 2, which is introduced by Yoon-Cheon-Kim (YCK) [30] and supports batch verifications of Type 3. Table 2 lists the main primitives required by the proposed signature scheme, Hess' scheme, Yi's scheme and YCK's scheme. Refer to Tables 1 and 2, it is obvious that the proposed scheme requires the shortest running time for extracting secret key. In the step of *sign*, both the proposed scheme and Hess' scheme require $1M_{G_1} + 1E$ which is faster than $3M_{G_1}$ in Yi's scheme and $1H_M + 3M_{G_1}$ in YCK's scheme. In the step of *verify*, the proposed scheme requires $1M_{G_1} + 1P + 1E$ which is more efficient than $1H_M + 1E + 2P$ in Hess's scheme, $1H_M + 1M_{G_1} + 2P$ in Yi's scheme and YCK's scheme. From the timings for the cryptographic primitives in Table 1, the verification of the proposed scheme makes an improvement of approximately 43% on Hess's scheme, 45% on Yi's scheme and YCK's scheme. We notice that Hess's scheme can reduce by one pairing computation in the step of *verify* when the same identities occur frequently [15], but two pairing computation is still necessary in the first verification. Therefore it is believable that the proposed scheme provides fastest verification in all the IBS schemes.

Although Yi's scheme doesn't require precomputation and provides the shortest signature, its signature scheme has to depend on some fixed elliptic curve [29]. However, the proposed scheme and Hess' scheme are not limited by this condition. In addition, by the technology of the point compression, the proposed scheme and Hess' scheme also provide the signature with the same size as Yi's scheme.

To verify the signatures on n distinct messages for n distinct signers, the batch verifications for Type 3 based on YCK's scheme require to compute $n+1$ pairings, n scalar multiplications and n *MapToPoint*. However, using the batch verifications on the proposed scheme, only two pairings, one exponentiation on G_2, $n-1$ multiplications on G_2, n scalar multiplications on G_1 are required. From Table 1, batch verification on YCK scheme requires about $(57n + 47)$ms, but the batch verification on the proposed scheme takes about $(7n{+}98)$ms. When n is a large number (e.g. $n \geq 100$), batch verification on the proposed scheme significantly reduces the verification time.

Finally, the recent research showed that the exponentiation operation on G_2 is time consuming when p and the embedding degree are large [16, 25]. Thus, we must notice that our proposed IBS scheme may not be more efficient than other schemes which do not require exponentiation operation.

7 Conclusion

In this paper, an efficient IBS scheme is introduced. Its security depends on a variant of k-CAA. This new IBS scheme improves the efficiency of extracting secret key and verifying signature by eliminating the special hash function called *MapToPoint* function. The results of the implementations indicate that the proposed scheme provides the most efficient key exaction and verification in all the IBS schemes from pairings. In particular, the efficiency of the verification is improved by at least 40% in some case. Furthermore, this new IBS scheme supports batch verifications which speeds up the verifications of multiple signatures. In the case of a lot of users and messages, the results show batch verifications on our scheme provide better performance than other IBS scheme. Furthermore, we also correct an error of an ID-based chameleon hash function in [32] such that the proposed IBS scheme is also suitable for collaborating on an efficient chameleon signature scheme with it. In the future, we will pay more attention to construct an IBS scheme without random oracles which is still an open problem.

References

[1] J. Arkko, T. Aura, J. Kempf, V. Mantyla, P. Nikander, and M. Roe, "Securing IPv6 neighbor discovery and router discovery," in *Proceedings of the ACM workshop on Wireless security (WiSe 2002)*, ACM Press, pp. 77-86, 2002.

[2] G. Ateniese and B. D. Medeiros, "Identity-based chameleon hash and applications", in *FC'04, Also*

in Cryptology ePrint Archive, Report 2003, vol. 167, 2004.

[3] P. S. L. M. Barreto, B. Libert, N. McCullagh, and J. Quisquater, "Efficient and provably-Secure identity-based signatures and signcryption from bilinear maps," in *Asiacrypt'05*, LNCS 3788, pp. 515-532, Springer-Verlag, 2005.

[4] M. Bellare, J. Garay, and T. Robin, "Fast batch verification for modular exponentiation and digital signatures,", in *Eurocrypt'98*, LNCS 1403, pp. 236-250, Springer-Verlag, 1998.

[5] I. Blake, G. Seroussi, and N. Smart, "Advances in elliptic curve cryptography," London Mathematical Society Lecture Note Series. Cambridge University Press, 2005.

[6] D.Boneh and M. Franklin, "Identity based encryption from the Weil pairing," in *Crypto'01*, LNCS 2139, pp. 213-229, Springer-Verlag, 2001.

[7] D. Boneh, B. Lynn and H. Shacham, "Short signature from Weil pairing," in *Asiacrypt'01*, LNCS 2248, pp. 514-532, Springer-Verlag, 2003.

[8] D. Boneh and X. Boyen, "Short signatures without random oracles", in *Eurocrypt'04*, LNCS 3027, pp. 56-73, Springer-Verlag, 2004.

[9] D. Boneh and X. Boyen, "Efficient selective-ID secure identity based encryption without random oracles," in *Eurocrypt'04*, LNCS 3027, pp. 223-238, Springer-Verlag, 2004.

[10] L. Chen and C. Kudla, "Identity based authenticated key agreement from pairings," in *Cryptology ePrint Archive, Report 2002*, vol. 184, 2002.

[11] J. C. Cha and J. H. Cheon, "An identity-based signature from gap Diffie-Hellman groups," in *PKC'03*, LNCS 2567, pp. 18-30, Springer-Verlag, 2003.

[12] D. Chaum and H. V. Antwerpen, "Undeniable signatures," in *Crypto'89*, LNCS 435, pp. 212-217, Springer-Verlag, 1989.

[13] X. Chen, F. Zhang, and K. Kim, "A new ID-based group signature scheme from bilinear pairings," in *WISA'03*, LNCS 2908, pp. 585-592, Springer-Verlag, 2003.

[14] C. Gentry and A. Silverberg, "Hierarchical ID-based cryptography," in *Asiacrypt'02*, LNCS 2501, pp. 548-566, Springer-Verlag, 2003.

[15] F. Hess, "Efficient identity based signature schemes based on pairings," in *SAC'02*, LNCS 2595, pp. 310-324, Springer-Verlag, 2003.

[16] N. Koblitz and A. Meneze, "Pairing-based cryptography at high security levels," in *10th IMA International Conference on Cryptography and Coding*, LNCS 3796, pp. 13-36, Springer-Verlag, 2005.

[17] H. Krawczyk and T. Rabin, "Chameleon signatures," in *Proceedings of Network and Distributed System Security Symposium (NDSS'00)*, pp. 143-154, 2000.

[18] B. Libert and J. j. Quisquater, "The exact security of an identity based signature and its applications," *Cryptology ePrint Archive, Report 2004*, vol. 102, 2004.

[19] S. Mitsunari, R. Sakai, and M. Kasahara, "A new traitor tracing," *IEICE Transactions on Fundamentals*, vol. E85-A, no. 2, pp. 481-484, 2002.

[20] K. G. Paterson, "ID-based signatures from pairings on elliptic curves," *Cryptology ePrint Archive, Report 2002*, vol. 003, 2002.

[21] K. G. Paterson and G. Price, "A comparison between traditional PKIs and identity-based cryptography," *Information Security Technical Report 8*, pp. 57-72, 2003.

[22] D. Pointcheval and J. Stern, "Security proofs for signature schemes," in *Eurocrypt'96*, LNCS 1992, pp. 387-398, Springer-Verlag, 1996.

[23] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *Journal of Cryptology*, vol. 13, no. 3, pp. 361-396, Springer-Verlag, 2000.

[24] R. Sakai, K. Ohgishi, and M. Kasahara. "Cryptosystems based on pairing," in *2000 Symposium on Cryptography and Information Security (SCIS'00)*, 2000.

[25] M. Scott, "Scaling security in pairing-based protocols," *Cryptology ePrint Archive, Report 2005*, vol. 139, 2005.

[26] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Crypto'84*, Santa Barbara, CA, pp.47-53, Aug. 1984.

[27] Shamus Software Ltd. Miracl: Multiprecision integer and rational arithmetic C/C++ library. (http://indigo.ie/~mscott/)

[28] N. Smart, "A ID-based authenticated key agreement protocol based on the Weil pairings," *Electronics Letters*, vol. 38, no. 13, pp. 630-632, 2002.

[29] X. Yi, "An identity-based signature scheme from the Weil pairing," *IEEE Communications Letters*, vol. 7, no. 2, pp. 76-78, 2003.

[30] H. Yoon, J. H. Cheon, and Y. Kim, "Batch verifications with ID-based signatures," in *ICISC'04*, LNCS 3506, pp. 223-248, Springer-Verlag, 2005.

[31] F. Zhang, R. Safavi-Naini, and W. Susilo,"ID-based chameleon hashes from bilinear pairings," *Cryptology ePrint Archive, Report 2003*, vol. 208, 2003.

[32] F. Zhang, R. Safavi-Naini and W. Susilo, "An efficient signature scheme from bilinear pairings and its applications," in *PKC'04*, LNCS 2947, pp. 277-290, Springer-Verlag, 2004.

[33] F. Zhang, X. Chen, W. Susilo, and Y. Mu, "A new short signature scheme without random oracles from bilinear pairings," *Cryptology ePrint Archive, Report'2005*, vol. 386, 2005.

Shi Cui Received his B.S. degree in Electronics and Information Science from Lanzhou University in 1998. He is currently a doctoral candidate in Information Security Center of School of School of Electronics and Electrical Engineering, Nanyang Technological University. His research interests are in the areas of public key cryptosystems.

Pu Duan Received his B.S. degree in Electronics and Information Science from Xi'an Jiaotong University in 2001. He is currently a doctoral candidate in Information Security Center of School of School of Electronics and Electrical Engineering, Nanyang Technological University. His research interests are in the areas of public key cryptosystems.

Choong Wah Chan Received his BSc, MSc and PhD in 1980, 1981 and 1984 respectively in United Kingdom. He also holds a PGDipTHE from National Institute of Education, NTU and a GDipBA from Singapore Institute of Management. He is currently an asspciate professor in School of School of Electronics and Electrical Engineering, Nanyang Technological University. He also holds the post of project leader of the ON-BOARD DATA HANDLING Subsystem of the DSO/SEC, NTU Satellite project. He has served as a Principal Consultant in Application Service Providers Centre (ASP Centre), NTU. His research interests are on copyright protection, elliptic curve cryptography, steganography, and information hiding in digital media.

Xiangguo Cheng Received his B.S. degree in Mathematics Science from Jilin University in 1992 and his M.S. degree in Applied Mathematics Science from Tongji University in 1998. He is currently a doctoral candidate under the instruction of Prof. Xinmei Wang at the State Key Laboratory of Integrated Services Network of Xidian University, P.R.China. His research interests are in the areas of information theory, Cryptography, and public key cryptosystems.