Key-Exposure Free Chameleon Hashing and Signatures Based on Discrete Logarithm Systems

Xiaofeng Chen, Fangguo Zhang, Haibo Tian, Baodian Wei, and Kwangjo Kim

[1] School of Information Science and Technology,
Sun Yat-sen University, Guangzhou 510275, P.R.China
{isschxf,isszhfg,tianhb,weibd}@mail.sysu.edu.cn
[2] International Research center for Information Security (IRIS)
Information and Communications University(ICU), Taejon 305-714, KOREA
kkj@icu.ac.kr

Abstract. Chameleon signatures are based on well established hash-and-sign paradigm, where a *chameleon hash function* is used to compute the cryptographic message digest. Chameleon signatures simultaneously provide the properties of non-repudiation and non-transferability for the signed message. However, the initial constructions of chameleon signatures suffer from the problem of key exposure: the signature forgery results in the signer recovering the recipient's trapdoor information, *i.e.*, the private key. This creates a strong disincentive for the recipient to forge signatures, partially undermining the concept of non-transferability. Recently, some specific constructions of key-exposure free chameleon hashing are presented, based on RSA or pairings, using the idea of "Customized Identities".

In this paper, we propose the first key-exposure free chameleon hash scheme based on discrete logarithm systems, without using the gap Diffile-Hellman groups. Moreover, one distinguished advantage of the resulting chameleon signature scheme is that the property of "message hiding" or "message recovery" can be achieved freely by the signer. Another main contribution in this paper is that we propose the first identity-based chameleon hash scheme without key exposure, which gives a positive answer for the open problem introduced by Ateniese and de Mederious in 2004.

Key words: Chameleon hashing, Gap Diffie-Hellman group, Key exposure.

1 Introduction

Chameleon signatures, introduced by Krawczyk and Rabin [20], are based on well established hash-and-sign paradigm, where a chameleon hash function is used to compute the cryptographic message digest. A chameleon hash function is a trapdoor one-way hash function, which prevents everyone except the holder of the trapdoor information from computing the collisions for a randomly given input. Chameleon signatures simultaneously provide non-repudiation and non-transferability for the signed message as undeniable signatures [3, 10–12, 16–19] do, but the former allows for simpler and more efficient realization than the latter. In particular, chameleon signatures are non-interactive and less complicated. More precisely, the signer can generate the chameleon signature without interacting with the designated recipient, and the recipient will be able to verify the signature without the collaboration of the signer. On the other hand, if presented with a forged signature, the signer can deny its validity by only revealing some certain values. That is, the forged-signature denial protocol is also non-interactive. Besides, since the chameleon signatures are based on well established hash-and-sign paradigm, it provides more generic and flexible constructions.

One limitation of the original chameleon signature scheme is that signature forgery results in the signer recovering the recipient's trapdoor information, *i.e.*, private key. The signer then can use this information to deny *other* signatures given to the recipient. In the worst case, the signer could collaborate with other individuals to invalidate any signatures which were designated to be verified by the same public key. This will create a strong disincentive for the recipient to forge signatures and thus weakens the property of non-transferability.

Ateniese and de Mederious [1] firstly addressed the key exposure problem of chameleon hashing and introduced the idea of identity-based chameleon hashing to solve this problem. Due to the distinguished property of identity-based system, the signer can sign a message to an intended recipient, without having to first retrieve the recipient's certificate. Moreover, the signer uses a different public key (corresponding a different private key) for each transaction with a recipient, so that signature forgery only results in the signer recovering the trapdoor information associated to a single transaction. Therefore, the signer will not be capable of denying signatures on any message in other transactions. We argue that this idea only provides a partial solution for the problem of key exposure since the recipient's public key is changed for each transaction.[1]

Chen et al. [15] proposed the first full construction of a key-exposure free chameleon hash function in the gap Diffie-Hellman (GDH) groups with bilinear pairings. Ateniese and de Mederious [2] then presented three key-exposure free chameleon hash schemes, two based on the RSA assumption (the first constructions without using pairings), as well as a new construction based on pairings. As pointed out by Ateniese and de Mederious, the single-trapdoor commitment schemes are not sufficient for the construction of key-exposure free chameleon hashing and the double-trapdoor mechanism can either be used to construct an identity-based chameleon hash scheme or a key-exposure free one, but not both. Therefore, an interesting open problem is whether there is an efficient construction for identity-based chameleon hash function without key exposure [2].

All of the existing key-exposure free chameleon hash schemes based on the discrete logarithm systems can only be constructed in the setting of GDH groups with bilinear pairings. Are there efficient (discrete-logarithm-based) constructions for key-exposure free chameleon hash schemes without using the GDH groups? To the best of our knowledge, it seems that there is no research work on this problem.

[1] A trivial solution for the key exposure problem is that the signer changes his key pair frequently in the chameleon signature scheme. However, it is only meaningful in theoretical sense because the key distribution problem arises simultaneously.

Our Contribution. In this paper, we propose some efficient constructions for key-exposure free chameleon hash schemes in the discrete logarithm systems. Our contribution is three folds:

1. We proposed a new key-exposure free chameleon hash scheme in the GDH groups. Compared with the existing schemes in the GDH groups [2, 15], the proposed chameleon hash scheme is not only based on the weaker assumption, but also more efficient in both hashing computation and collision computation.

2. We propose the first discrete-logarithm-based key-exposure free chameleon hash scheme without using the GDH groups. One distinguished advantage of the resulting chameleon signature scheme is that the property of "message hiding" or "message recovery" can be achieved freely by the signer.

3. We propose the first identity-based chameleon hash scheme without key exposure, which gives a positive answer for the open problem introduced by Ateniese and de Mederious in 2004.

Organization. The rest of the paper is organized as follows: Some preliminaries are given in Section 2. The definitions associated with chameleon hashing and signatures are introduced in Section 3. The proposed key exposure freeness chameleon hash and signature schemes in the GDH groups and non-GDH groups are given in Section 4 and Section 5, respectively. The proposed identity-based chameleon hash scheme without key exposure is given in Section 6. Finally, conclusions will be made in Section 7.

2 Preliminaries

In this section, we first introduce some well-known number-theoretic problems in the discrete logarithm systems. We then present two proof systems for knowledge of discrete logarithms.

2.1 Number-Theoretic Problems

Let $\mathbb{G}$ be a cyclic multiplicative group generated by g with the prime order q. We introduce the following problems in $\mathbb{G}$.

- Discrete Logarithm Problem (DLP): Given two elements g and h, to find an integer $a \in \mathbb{Z}_q^*$, such that $h = g^a$ whenever such an integer exists.
- Computation Diffie-Hellman Problem (CDHP): Given (g, g^a, g^b) for $a, b \in \mathbb{Z}_q^*$, to compute g^{ab}.
- Decision Diffie-Hellman Problem (DDHP): Given (g, g^a, g^b, g^c) for $a, b, c \in \mathbb{Z}_q^*$, to decide whether $c \equiv ab \bmod q$.

It is proved that the CDHP and DDHP are not equivalent in the GDH groups. More precisely, we call $\mathbb{G}$ a GDH group if the DDHP can be solved in polynomial time but there is no polynomial time algorithm to solve the CDHP with non-negligible probability. Such groups can be found in supersingular elliptic curves or hyperelliptic curves over finite fields. For more details, see [4–6, 9, 21, 23, 25]. Moreover, we call $< g, g^a, g^b, g^c >$ a valid Diffie-Hellman tuple if $c \equiv ab \bmod q$.

2.2 Proofs of Knowledge

A prover with possession a secret number $x \in \mathbb{Z}_q$ wants to show a verifier that $x = \log_g y$ without exposing x, this is named the proof of knowledge of a discrete logarithm.

This proof of knowledge is basically a Schnorr signature [26] on message (g, y): The prover chooses a random number $r \in_R \mathbb{Z}_q$, and then computes $c = H(g, y, g^r)$, and $s = r - cx \mod q$, where $H : \{0,1\}^* \to \{0,1\}^k$ is a collision-resistant hash function. The verifier accepts the proof if and only if $c = H(g, y, g^s y^c)$.

Definition 1. *A pair $(c, s) \in \{0,1\}^k \times \mathbb{Z}_q$ satisfying $c = H(g, h, g^s y^c)$ is a proof of knowledge of a discrete logarithm of the element y to the base g.*

Similarly, we can define the proof of knowledge for the equality of two discrete logarithms: A prover with possession a secret number $x \in \mathbb{Z}_q$ wants to show that $x = \log_g u = \log_h v$ without exposing x.

Chaum and Pedersen [14] firstly proposed the proof as follows: The prover chooses a random number $r \in_R \mathbb{Z}_q$, and then computes $c = H(g, h, u, v, g^r, h^r)$, and $s = r - cx \mod q$, where $H : \{0,1\}^* \to \{0,1\}^k$ is a collision-resistant hash function. The verifier accepts the proof if and only if $c = H(g, h, u, v, g^s u^c, h^s v^c)$.

Definition 2. *A pair $(c, s) \in \{0,1\}^k \times \mathbb{Z}_q$ satisfying $c = H(g, h, u, v, g^s u^c, h^s v^c)$ is a proof of knowledge for the equality of two discrete logarithms of elements u, v with respect to the base g, h.*

Trivially, the verifier can efficiently decide whether $< g, u, h, v >$ is a valid Diffie-Hellman tuple with the pair (c, s).

3 Definitions

In this section, we introduce the definitions and properties of chameleon hashing and signatures [1, 20, 28].

3.1 Chameleon Hashing

A chameleon hash function is a trapdoor collision-resistant hash function, which is associated with a trapdoor/hash key pair (TK, HK). Anyone who knows the public key HK can efficiently compute the hash value for each input. However, there exists no efficient algorithm for anyone except the holder of the secret key TK, to find collisions for every given input. In the following, we present a formal definition of a chameleon hash scheme.

Definition 3. *A chameleon hash scheme consists of four efficient algorithms* $(\mathcal{PG}, \mathcal{KG}, \mathcal{H}, \mathcal{F})$*:*

- **System Parameters Generation** $\mathcal{PG}$**:** *A probabilistic polynomial-time algorithm that, on input a security parameter* k*, outputs the system parameters* SP*.*
- **Key Generation** $\mathcal{KG}$ **:** *A probabilistic polynomial-time algorithm that, on input the system parameters* SP*, outputs a trapdoor/hash key pair* (TK, HK)*.*
- **Hashing Computation** $\mathcal{H}$**:** *A probabilistic polynomial-time algorithm that, on input the hash key* HK*, a customized identity* I*,*[3] *a message* m*, and a random string* r*, outputs the hashed value* $h = Hash(I, m, r)$*. Note that* h *does not depend on* TK*.*
- **Collision Computation** $\mathcal{F}$**:** *A deterministic polynomial-time algorithm that, on input the trapdoor key* TK*, a message* m*, a random string* r*, and another message* $m' \neq m$*, outputs a string* r' *that satisfies*

  $$Hash(I, m', r') = Hash(I, m, r).$$

  Moreover, if r *is uniformly distributed in a finite space* $\mathcal{R}$*, then the distribution of* r' *is computationally indistinguishable from uniform in* $\mathcal{R}$*.*

A secure chameleon hashing scheme satisfies the following properties:

- **Collision resistance:** Without the knowledge of trapdoor key TK, there exists no efficient algorithm that, on input a message m, a random string r, and another message m', outputs a string r' that satisfy $Hash(I, m', r') = Hash(I, m, r)$, with non-negligible probability.
- **Semantic security:** For all pairs of messages m and m', the probability distributions of the random values $Hash(I, m', r)$ and $Hash(I, m, r)$ are computationally indistinguishable.
- **Key exposure freeness:** If a recipient has never computed a collision under I, then there is no efficient algorithm for an adversary to find a collision for a given chameleon hash value $Hash(I, m, r)$. This must remain true even if the adversary has oracle access to $\mathcal{F}$ and is allowed polynomially many queries on triples (I_j, m_j, r_j) of his choice, except that I_j is not allowed to equal the challenge I.

3.2 Chameleon Signatures

A chameleon signature is generated by digitally signing a chameleon hash value of the message. More precisely, we have the following definition:

Definition 4. *A chameleon signature scheme consists of the following efficient algorithms and a specific denial protocol:*

[3] A customized identity is actually a label for each transaction. For example, we can let $I = ID_S||ID_R||ID_T$, where ID_S, ID_R, and ID_T denote the identity of the signer, recipient, and transaction, respectively [1].

– **System Parameters Generation** $\mathcal{PG}$**:** *A probabilistic polynomial-time algorithm that, on input a security parameter* k*, outputs the system parameters* SP*.*
– **Key Generation** $\mathcal{KG}$ **:** *A probabilistic polynomial-time algorithm that, on input the system parameters* SP*, outputs a trapdoor/hash key pair* (TK, HK) *and a signing/verification key pair* (sk, vk)*.*
– **Signature Generation** $\mathcal{SG}$**:** *A probabilistic polynomial-time algorithm that, on input the hash key* HK*, the signing key* sk*, a customized identity* I*, a message* m*, and a random string* r*, outputs a signature* σ *on the chameleon hash value* $h = Hash(I, m, r)$*.*
– **Signature Verification** $\mathcal{SV}$**:** *A deterministic polynomial-time algorithm that, on input the hash key* HK*, the verification key* vk*, a customized identity* I*, a message* m*, a random string* r*, and a signature* σ*, outputs a verification decision* $b \in \{0, 1\}$*.*
– **Denial Protocol** $\mathcal{DP}$**:** *A non-interactive protocol between the signer and the judge. Given a chameleon signature* (σ, r) *on the message* m*, the signer provides the judge a valid collision* (m', r') *and some auxiliary information* Σ*. If and only if* $m \neq m'$ *and* Σ *is valid, the judge claims that the signature* σ *on the message* m *is a forgery.*

A secure chameleon signature scheme should satisfy the properties [1, 15, 20]:

– **Unforgeability**: No party can produce a valid chameleon signature not previously generated by the signer. Also, the recipient can only produce a forgery of a chameleon signature previously generated by the signer.
– **Non-transferability**: The recipient can not convince a third party that the signer indeed generated a signature on a certain message, thus the signature is not universal verifiable.
– **Non-repudiation**: The signer cannot deny legitimate signature claims.
– **Deniability**: The signer can deny a forgery of the signature.
– **Message hiding**: The signer does not have to reveal the original message to deny the validity of a forgery.
– **Message recovery** (or **Convertibility**): A variant of the chameleon signature can be transformed into a regular signature by the signer.

4 Constructions in the GDH Groups

In this section, we present an efficient construction of chameleon hashing without key exposure in the GDH groups.

4.1 The Proposed Chameleon Hash Scheme

– **System Parameters Generation** $\mathcal{PG}$**:** Let $\mathbb{G}$ be a GDH group generated by g, whose order is a prime q. Let $H : \{0, 1\}^* \rightarrow \mathbb{G}^*$ be a full-domain collision-resistant hash function. The system parameters are $SP = \{\mathbb{G}, q, g, H\}$.

- **Key Generation** $\mathcal{KG}$ **:** Any user randomly chooses an integer $x \in_R \mathbb{Z}_q^*$ as his trapdoor key, and publishes his hash key $y = g^x$. The validity of y can be ensured by a certificate issued by a trusted certification authority.
- **Hashing Computation** $\mathcal{H}$**:** On input the hash key y, a customized identity I, let $h = H(y, I)$. Chooses a random integer $a \in_R \mathbb{Z}_q^*$, and computes $r = (g^a, y^a)$. Our proposed chameleon hash function is defined as

$$\mathcal{H} = \mathrm{Hash}(I, m, r) = g^a h^m.$$

- **Collision Computation** $\mathcal{F}$**:** For any valid hash value $\mathcal{H}$, the algorithm $\mathcal{F}$ can be used to compute a hash collision with the trapdoor key x as follows:

$$\mathcal{F}(\mathcal{H}, x, I, m, r, m') = r' = (g^{a'}, y^{a'}),$$

  where $g^{a'} = g^a h^{m-m'}$ and $y^{a'} = y^a h^{x(m-m')}$.

Note that

$$\mathrm{Hash}(I, m', r') = g^{a'} h^{m'} = g^a h^{m-m'} h^{m'} = g^a h^m = \mathrm{Hash}(I, m, r)$$

and $< g, y, g^{a'}, y^{a'} >$ is a valid Diffie-Hellman tuple. Therefore, the forgery is successful. Moreover, if r is uniformly distributed then the distribution of r' is computationally indistinguishable from uniform.

Theorem 1. *The proposed chameleon hash scheme is collision resistance under the assumption that the CDHP in* $\mathbb{G}$ *is intractable.*

Proof. Assume to the contrary, that there exists a polynomial time algorithm $\mathcal{A}$, with a non-negligible probability, that outputs two pairs (m, r) and (m', r') which satisfy $\mathrm{Hash}(I, m', r') = \mathrm{Hash}(I, m, r)$, *i.e.*, $g^{a'} h^{m'} = g^a h^m$, we can compute $h^x = (y^{a'}/y^a)^{(m-m')^{-1}}$ efficiently. This is equivalent to solve the CDHP in $\mathbb{G}$. □

Theorem 2. *The proposed chameleon hash scheme is semantically secure.*

Proof. Given a value $\mathcal{H}$, a customized identity I, and any message m, there exists exactly one string r such that $\mathcal{H} = \mathrm{Hash}(I, m, r)$. □

Theorem 3. *The proposed chameleon hash scheme is key-exposure free.*

Proof. Even if the adversary has oracle access to $\mathcal{F}$ and is allowed polynomially many queries on triples $(I_j, m_j, g^{a_j}, y^{a_j})$ of his choice, there is no efficient algorithm for him to find a collision of the hash value $\mathcal{H} = \mathrm{Hash}(I, m, g^a, y^a)$ where $I \neq I_j$. Note that h^x is a GDH signature on message I [5], and computing collisions is equivalent to breaking the signature scheme. However, the GDH signature scheme is proved to be secure against existential forgery on adaptive chosen-message attacks in the random oracle model. In other words, even if the adversary has obtained polynomially many GDH signatures h_j^x on message I_j, he can not forge a signature h^x on message $I \neq I_j$. □

4.2 The Proposed Chameleon Signature Scheme

There are two users, a signer S and a recipient R, in our signature scheme. When dispute occurs, a judge J is involved in the scheme.

- **System Parameters Generation $\mathcal{PG}$:** Let $\mathbb{G}$ be a GDH group generated by g, whose order is a prime q. Let $H : \{0,1\}^* \to \mathbb{G}^*$ be a full-domain collision-resistant hash function. The system parameters are $SP = \{\mathbb{G}, q, g, H\}$.
- **Key Generation $\mathcal{KG}$:** S randomly chooses an integer $x_S \in_R \mathbb{Z}_q^*$ as his signing key, and publishes his verification key $y_S = g^{x_s}$. Similarly, R randomly chooses an integer $x_R \in_R \mathbb{Z}_q^*$ as his trapdoor key, and publishes his hash key $y_R = g^{x_R}$.
- **Signature Generation $\mathcal{SG}$:** Suppose the message to be signed is m. S randomly chooses an integer $a \in_R \mathbb{Z}_q^*$, and computes the chameleon hash value $\mathcal{H} = g^a h^m$, where $h = H(y_R, I)$ and I is a customized identity. Assume SIGN is any secure signature scheme. The signature σ for message m consists of $(m, g^a, y_R^a, \mathrm{SIGN}_{x_S}(\mathcal{H}))$.
- **Signature Verification $\mathcal{SV}$:** Given a signature σ, R first verifies whether the equation $(g^a)^{x_R} = y_R^a$ holds.[3] If the verification fails, he rejects the signature; else, he computes the chameleon hash value $\mathcal{H} = g^a h^m$ and verifies the validity of $\mathrm{SIGN}_{x_S}(\mathcal{H})$ with the verification key y_S.
- **Denial Protocol $\mathcal{DP}$:** When dispute occurs, *i.e.*, R provides a signature σ=$(m^*, g^{a^*}, y_R^{a^*}, \mathrm{SIGN}_{x_S}(\mathcal{H}))$ to J. If either $< g, y_R, g^{a^*}, y_R^{a^*} >$ is not a valid Diffie-Hellman tuple or $\mathrm{SIGN}_{x_S}(\mathcal{H})$ is invalid, J rejects it. Otherwise, J summons S to accept/deny the claim. If S wants to accept the signature, he just confirms to J this fact. Otherwise, he provides a collision for the chameleon hash function as follows:
  - If S wants to achieve the property of "message hiding", he provides J a collision $(m', g^{a'}, y_R^{a'})$. If and only if $m^* \neq m'$, $< g, y_R, g^{a'}, y_R^{a'} >$ is a valid Diffie-Hellman tuple, and $\mathcal{H} = g^{a'} h^{m'}$, then J can be convinced that R forged the signature on message m^*.
  - If S wants to achieve the property of "message recovery", he provides the tuple (m, g^a, y_R^a, Σ) as the collision, where Σ is a non-interactive proof of knowledge of the discrete logarithm $a = \log_g g^a$. If and only if $m^* \neq m$, $< g, y_R, g^a, y_R^a >$ is a valid Diffie-Hellman tuple, $\mathcal{H} = g^a h^m$, and Σ is valid, then J can be convinced that R forged the signature on message m^* and S only generated a valid signature on message m.

Different from the basic chameleon signature schemes [1, 20], the proposed chameleon signature scheme has the following distinguishing advantages:

1. In the previous chameleon signature schemes, the customized identity I and the identity of the recipient ID_R must be explicitly committed to the signature. While in our scheme, this is not required since no one knows the discrete logarithm of the element h to the base g.

[3] If the equation $(g^a)^{x_R} = y_R^a$ holds, then R can be convinced that $< g, y_R, g^a, y_R^a >$ is a valid Diffie-Hellman tuple.

2. Another distinguishing advantage of our scheme is that the signer can efficiently prove which message was the original one if he desires. This is due to the following observations: Firstly, no one can provide a proof of knowledge of the discrete logarithm $a' = \log_g g^{a'}$ for any collision $g^{a'} = g^a h^{m-m'}$; Secondly, only S can provide a proof of knowledge of the discrete logarithm $a = \log_g g^a$ for the original input g^a.
   On the other hand, the enhanced schemes [1, 20] can be converted into universally verifiable instances. The trick is that the signer encrypts the message using a semantically secure probabilistic encryption scheme ENC and then includes the ciphertext in the signature. However, as noted in [1], this solution does not provide the recipient with a mechanism for adjudicated convertibility, because the recipient has no guarantee that the signer has encrypted the correct information during the signing step.

4.3 Security Analysis

Theorem 4. *The proposed chameleon signature scheme satisfies the properties of unforgeability, non-transferability, non-repudiation, deniability, message hiding,* and *key exposure freeness.*

Proof. We prove the proposed chameleon signature scheme satisfies the above properties one by one.

- **Unforgeability**: No third party can produce a valid chameleon signature which has not been previously generated by the signer, as this requires either to break the underlying signature scheme SIGN, or find a valid collision of the chameleon hash function $\mathcal{H}$. Also, it is trivial that the recipient can only produce a forgery of a chameleon signature previously generated by the signer. However, it is meaningless since the judge can detect this forgery after the signer provides a different collision.
- **Non-transferability**: Note that the semantic security of a chameleon hashing scheme implies the non-transferability of the corresponding chameleon signature scheme [1]. Therefore, the recipient cannot transfer a signature of the signer to convince any third party.
- **Non-repudiation**: Given a valid signature $\sigma = (m, g^a, y_R^a, \mathrm{SIGN}_{x_S}(\mathcal{H}))$, the signer cannot generate a valid hash collision $(m', g^{a'}, y_R^{a'})$ which satisfies $\mathcal{H} = \mathrm{Hash}(I, m', g^{a'}, y_R^{a'})$ and $m \neq m'$ because it is equivalent to computing the CDHP in $\mathbb{G}$.
- **Deniability**: It is ensured by the denial protocol.
- **Message hiding**: Given a collision (m, g^a, y_R^a) and $(m^*, g^{a^*}, y_R^{a^*})$, though the trapdoor key x is never divulged, the signer can compute the *ephemeral* trapdoor key h^x. Then the signer can provide any other collision $(m', g^{a'}, y_R^{a'})$ to ensure the confidentiality of the original message m, where $g^{a'} = g^a h^{m-m'}$, $y_R^{a'} = y_R^a (h^x)^{m-m'}$.

– **Message recovery**: Note that (only) S can provide a proof of knowledge of the discrete logarithm $a = \log_g g^a$ (only) for the original input g^a. Therefore, any verifier can be convinced that the original message to be signed is m.

4.4 Comparison

Compared with the existing two key-exposure free chameleon hash schemes in the GDH groups [2, 15], the proposed chameleon hash scheme is a little more efficient in both hashing computation and collision computation. Moreover, the security of the scheme [2] is equivalent to the q-Strong Diffie-Hellman Problem (q-SDHP), while the security of our proposed scheme is equivalent to the CDHP, which is harder than the q-SDHP for any q.

In the proposed chameleon signature scheme, both the signature verification and the denial protocol are non-interactive, so it is more efficient and simple than undeniable signature schemes. Moreover, compared with two previous chameleon signature schemes in the GDH groups [2, 15], our signature scheme provides more efficient and explicit convertibility.

Table 1 and Table 2 present the comparison between our scheme and two previous schemes. We denote by M the exponentiation in $\mathbb{G}$, by m the multiplication in $\mathbb{G}$, and by I the inversion in $\mathbb{G}$. We also denote by $C(S)$, $C(V)$, and $C(E)$ the computation cost of signing, verifying in scheme SIGN and encrypting in scheme ENC, respectively. We omit other operations such as hash and the multiplication in $\mathbb{Z}_q$ in all schemes.

	Scheme [2]	Scheme [15]	Our Scheme
Mathematical Assumption	q-SDHP	CDHP	CDHP
Hashing Computation	$4M + 2m$	$3M + 2m$	$3M + 1m$
Collision Computation	$2M + 2m$	$2M + 2m$	$2M + 2m$

Table 1. Comparison with two previous chameleon hash schemes

	Scheme [2]	Scheme [15]	Our Scheme
Signature Generation	$4M + 2m + 1C(S)$	$3M + 2m + 1C(S)$	$3M + 1m + 1C(S)$
Signature Verification	$2M + 1m + 1C(V)$	$2M + 2m + 1C(V)$	$2M + 1m + 1C(V)$
Denial Protocol (Message Hiding)	$2M + 3m + 1I$	$2M + 3m + 1I$	$2M + 3m + 1I$
Denial Protocol (Message Recovery)	$1C(E)$	$1C(E)$	$1M$

Table 2. Comparison with two previous chameleon signature schemes

5 Constructions in the Non-GDH Groups

In this section, we propose a construction of key exposure freeness chameleon hashing in the non-GDH groups, *e.g.*, the multiplicative group of finite fields.

5.1 Main Idea

In the non-GDH groups, there is no polynomial time algorithm to solve the DDHP with non-negligible probability. Therefore, given a tuple $< g, g^a, g^b, g^c >$, no one is allowed to use the decisional Diffie-Hellman (DDH) oracle to check whether it is a valid Diffie-Hellman tuple.

However, as we mentioned above, the proof of knowledge for the equality of two discrete logarithms can substitute the DDH oracle. This is the main trick to design key exposure freeness chameleon hash scheme in the non-GDH groups. We explain it in more details as below.

The chameleon hash scheme in the non-GDH groups is almost the same as the one in the GDH groups. The only difference is the way to verify the validity of a Diffie-Hellman tuple. Given the original input (g^a, y^a) in the GDH groups, anyone can easily check that $< g, y, g^a, y^a >$ is a valid Diffie-Hellman tuple using the DDH oracle. While in the non-GDH groups, on one except the holder of the trapdoor key x can verify the validity of the Diffie-Hellman tuple $< g, y, g^a, y^a >$. However, the holder can check whether the equation $(g^a)^x = y^a$ holds using the trapdoor key x. If the equation holds, then (g^a, y^a) is a valid input of the chameleon hashing. Moreover, the holder with x can provide a proof of knowledge for the equality of two discrete logarithms, *i.e.*, $x = \log_g y = \log_{g^a} y^a$, to convince any third party of the fact.

5.2 The Proposed Chameleon Hash Scheme

- **System Parameters Generation $\mathcal{PG}$:** Let $\mathbb{G}$ be a multiplicative group generated by g, whose order is a prime q. Let $H : \{0,1\}^* \to \mathbb{G}^*$ be a full-domain collision-resistant hash function. The system parameters are $SP = \{\mathbb{G}, q, g, H\}$.
- **Key Generation $\mathcal{KG}$:** Any user randomly chooses an integer $x \in_R \mathbb{Z}_q^*$ as his trapdoor key, and publishes his hash key $y = g^x$. The validity of y can be ensured by a certificate issued by a trusted certification authority.
- **Hashing Computation $\mathcal{H}$:** On input the hash key y, a customized identity I, let $h = H(y, I)$. Chooses a random integer $a \in_R \mathbb{Z}_q^*$, and computes $r = (g^a, y^a)$. Our proposed chameleon hash function is defined as

$$\mathcal{H} = \mathrm{Hash}(I, m, r) = g^a h^m.$$

- **Collision Computation $\mathcal{F}$:** For any valid hash value $\mathcal{H}$, the algorithm $\mathcal{F}$ can be used to compute a hash collision with the trapdoor key x as follows:

$$\mathcal{F}(\mathcal{H}, x, I, m, r, m') = r' = (g^{a'}, y^{a'}),$$

  where $g^{a'} = g^a h^{m-m'}$ and $y^{a'} = y^a h^{x(m-m')}$.

Note that $\mathrm{Hash}(I, m', r') = \mathrm{Hash}(I, m, r)$. Also, for any collision r', the holder of the trapdoor key x can convince any third party that $< g, y, g^{a'}, y^{a'} >$ is a valid Diffie-Hellman tuple, using a proof of knowledge for the equality of two discrete logarithms, *i.e.*, $\log_g y = \log_{g^{a'}} y^{a'}$. In particular, it also holds for the original input (g^a, y^a). Therefore, the forgery is successful. Besides, if r is uniformly distributed then the distribution of r' is computationally indistinguishable from uniform.

Theorem 5. *The construction above is a secure chameleon hash scheme under the assumption that the CDHP in* $\mathbb{G}$ *is intractable.*

Proof. The proof for the properties of collision resistance and semantic security is the same as that of theorem 1. In the following, we only focus on the key exposure freeness.

Note that even if the adversary has obtained polynomially many signatures h_j^x on message I_j, he can not forge a signature h^x on message $I \neq I_j$, otherwise the full domain hash (FDH) [8, 13] variant of Chaum's undeniable signature scheme can be broken. However, Ogata et al. [24] showed that the unforgeability of the FDH variant of Chaum's scheme with non-interactive zero-knowledge proof confirmation and disavowal protocols is equivalent to the CDHP. Therefore, even if the adversary has oracle access to $\mathcal{F}$ and is allowed polynomially many queries on triples $(I_j, m_j, g^{a_j}, y^{a_j})$ of his choice, there is no efficient algorithm for him to find a collision of the hash value $\mathcal{H} = \mathrm{Hash}(I, m, g^a, y^a)$ where $I \neq I_j$. $\square$

5.3 The Proposed Chameleon Signature Scheme

There are two users, a signer S and a recipient R, in our signature scheme. When dispute occurs, a judge J is involved in the scheme.

- **System Parameters Generation $\mathcal{PG}$:** Let $\mathbb{G}$ be a multiplicative group generated by g, whose order is a prime q. Let $H : \{0,1\}^* \rightarrow \mathbb{G}^*$ be a full-domain collision-resistant hash function. The system parameters are $SP = \{\mathbb{G}, q, g, H\}$.
- **Key Generation $\mathcal{KG}$:** S randomly chooses an integer $x_S \in_R \mathbb{Z}_q^*$ as his signing key, and publishes his verification key $y_S = g^{x_s}$. Similarly, R randomly chooses an integer $x_R \in_R \mathbb{Z}_q^*$ as his trapdoor key, and publishes his hash key $y_R = g^{x_R}$.
- **Signature Generation $\mathcal{SG}$:** Suppose the message to be signed is m. S randomly chooses an integer $a \in_R \mathbb{Z}_q^*$, and computes the chameleon hash value $\mathcal{H} = g^a h^m$, where $h = H(y_R, I)$ and I is a customized identity. Assume SIGN is any secure signature scheme. The signature σ for message m consists of $(m, g^a, y_R^a, \mathrm{SIGN}_{x_S}(\mathcal{H}))$.
- **Signature Verification $\mathcal{SV}$:** Given a signature σ, R first verifies whether the equation $(g^a)^{x_R} = y_R^a$ holds. If the verification fails, he rejects the signature; else, he computes the chameleon hash value $\mathcal{H} = g^a h^m$ and verifies the validity of $\mathrm{SIGN}_{x_S}(\mathcal{H})$ with the verification key y_S.

– **Denial Protocol $\mathcal{DP}$:** When dispute occurs, *i.e.*, R provides J a signature σ=$(m^*, g^{a^*}, y_R^{a^*}, \mathrm{SIGN}_{x_S}(\mathcal{H}))$ and a non-interactive proof of knowledge Π^* for the equality of two discrete logarithms that $x_R = \log_g y_R = \log_{g^{a^*}} y_R^{a^*}$. If either $\mathrm{SIGN}_{x_S}(\mathcal{H})$ or Π^* is invalid, J rejects it. Otherwise, J summons S to accept/deny the claim. If S wants to accept the signature, he just confirms to J this fact. Otherwise, he provides a collision for the chameleon hash function as follows:
  - If S wants to achieve the property of "message recovery", he provides J the tuple (m, g^a, y_R^a, Π) as a collision, where Π is a non-interactive proof of knowledge for the equality of two discrete logarithms that $\log_g g^a = \log_{y_R} y_R^a$. If and only if $m^* \neq m$, $\mathcal{H} = g^a h^m$, and Π is valid, then J can be convinced that R forged the signature on message m^* and S only generated a valid signature on message m.
  - If S wants to achieve the property of "message hiding", he provides J the tuple $(g^a, y_R^a, \Sigma, \Pi)$ as a collision, where Σ is a non-interactive proof of knowledge of a discrete logarithm that $m = \log_h \mathcal{H}/g^a$, and Π is a non-interactive proof of knowledge for the equality of two discrete logarithms that $\log_g g^a = \log_{y_R} y_R^a$. If and only if $g^{a^*} \neq g^a$, and Σ and Π are both valid, then J can be convinced that R forged the signature on message m^* and the original message m is still confidential.

Remark 1. For any collision $(g^{a^*}, y_R^{a^*})$, R can provide a proof of knowledge that $\log_g y_R = \log_{g^{a^*}} y_R^{a^*}$, which is also holds even when $a = a^*$. That is, the original input (g^a, y_R^a) is totally indistinguishable with any collision $(g^{a^*}, y_R^{a^*})$. Besides, only S can provide a proof of knowledge that $\log_g g^a = \log_{y_R} y_R^a$, and no one can provide a proof of knowledge that $\log_g g^{a^*} = \log_{y_R} y_R^{a^*}$ when $a \neq a^*$. Therefore, S can efficiently prove which message was the original one if he desires.

Remark 2. In the proposed chameleon signature scheme, both the signature verification and the denial protocol are non-interactive, so it is more efficient and simple than undeniable signature schemes.

Compared with our key-exposure free chameleon signature scheme based on GDH groups in section 4.2, the proposed scheme is as efficient as in the signature generation and verification algorithms. While in the denial protocol, the proposed scheme requires a (very) little more computation and communication cost for the non-interactive proofs of knowledge. We argue that these proofs of knowledge requires at most 2 modular exponentiation operations and about $2q$ bits storage. Therefore, the proposed chameleon signature scheme is much efficient for the real applications.

5.4 Security Analysis

Theorem 6. *The proposed chameleon signature scheme satisfies the properties of unforgeability*, *non-transferability*, *non-repudiation*, *deniability*, *message hiding*, and *key exposure freeness.*

Proof. We prove the proposed chameleon signature scheme satisfies the above properties one by one.

- **Unforgeability**: No third party can produce a valid chameleon signature which has not been previously generated by the signer, as this requires either to break the underlying signature scheme SIGN, or find a valid collision of the chameleon hash function $\mathcal{H}$. Also, it is trivial that the recipient can only produce a forgery of a chameleon signature previously generated by the signer. However, it is meaningless since the judge can detect this forgery after the signer provides a different collision.
- **Non-transferability**: The semantic security of the proposed chameleon hash scheme implies the non-transferability of the resulting chameleon signature scheme.
- **Non-repudiation**: Given a valid signature $\sigma = (m, g^a, y_R^a, \mathrm{SIGN}_{x_S}(\mathcal{H}))$, the signer cannot generate a valid hash collision $(m', g^{a'}, y_R^{a'})$ which satisfies $\mathcal{H} = \mathrm{Hash}(I, m', g^{a'}, y_R^{a'})$ and $m \neq m'$ because it is equivalent to computing the CDHP in $\mathbb{G}$.
- **Deniability**: It is ensured by the denial protocol.
- **Message hiding**: Since Σ is a proof of knowledge of a discrete logarithm that $m = \log_h \mathcal{H}/g^a$, the information for original signed message m is never revealed.
- **Message recovery**: Note that only S can provide a proof of knowledge that $\log_g g^a = \log_{y_R} y_R^a$, and no one can provide a proof of knowledge that $\log_g g^{a^*} = \log_{y_R} y_R^{a^*}$ when $a^* \neq a$. Therefore, any verifier can be convinced that the original message to be signed is m.

6 Identity-based Key-exposure Free Chameleon Hashing

In this section, we first propose an identity-based key-exposure free chameleon hash scheme based on bilinear pairings, which still follows the above construction while using an identity-based proof of knowledge for the equality of two discrete logarithms [7].

Let $\mathbb{G}_1$ be a GDH group generated by P, whose order is a prime q, and $\mathbb{G}_2$ be a cyclic multiplicative group of the same order q. A bilinear pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. Recently, the bilinear pairings play an important role in designing identity-based cryptographic schemes. The concept of identity-based public key systems, introduced by Shamir in 1984 [27], allows a user to use his identity ID as the public key, and a trusted third party, called Private Key Generator (PKG), calculates the private key S_{ID} for the user.

The identity-based proof of knowledge for the equality of two discrete logarithms, first introduced by Baek and Zheng [7] from bilinear pairings. Define $g = e(P, P)$, $u = e(P, S_{ID})$, $h = e(L, P)$ and $v = e(L, S_{ID})$, where P and L are independent points of $\mathbb{G}_1$. The following non-interactive protocol presents a proof of knowledge that $\log_g u = \log_h v$: The prover chooses a random number

$r \in_R \mathbb{Z}_q$, and then computes $c = H(g, h, u, v, g^r, h^r)$, and $S = rP - cS_{ID}$, where $H : \{0,1\}^* \to \{0,1\}^k$ is a collision-resistant hash function. The verifier accepts the proof if and only if $c = H(g, h, u, v, e(P, S)u^c, e(L, S)v^c)$.

6.1 The Proposed Identity-based Chameleon Hash Scheme

- **System Parameters Generation $\mathcal{PG}$:** Let $\mathbb{G}_1$ be a GDH group generated by P, whose order is a prime q, and $\mathbb{G}_2$ be a cyclic multiplicative group of the same order q. A bilinear pairing is a map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$. Let $H : \{0,1\}^* \to \mathbb{G}_1$ be a full-domain collision-resistant hash function. PKG picks a random integer $s \in_R \mathbb{Z}_q^*$ and computes $P_{pub} = sP$. The system parameters are $SP = \{\mathbb{G}_1, \mathbb{G}_2, q, P, P_{pub}, H\}$.
- **Key Generation $\mathcal{KG}$:** Given an identity string ID, computes the trapdoor key $S_{ID} = sH(ID) = sQ_{ID}$.
- **Hashing Computation $\mathcal{H}$:** On input the hash key ID, a customized identity I, let $C = H(I)$. Chooses a random integer $a \in_R \mathbb{Z}_q^*$, and computes $(aP, e(aP_{pub}, Q_{ID}))$. Our proposed chameleon hash function is defined as

  $$\mathcal{H} = \mathrm{Hash}(I, m, aP, e(aP_{pub}, Q_{ID})) = aP + mC.$$

  Note that the holder of the trapdoor key S_{ID} can check whether the equation $e(aP, S_{ID}) = e(aP_{pub}, Q_{ID})$ holds.
- **Collision Computation $\mathcal{F}$:** For any valid hash value $\mathcal{H}$, the algorithm $\mathcal{F}$ can be used to compute a hash collision with the trapdoor key S_{ID} as follows:

  $$\mathcal{F}(\mathcal{H}, S_{ID}, I, m, aP, e(aP_{pub}, Q_{ID}), m') = (a'P, e(a'P_{pub}, Q_{ID})),$$

  where

  $$a'P = aP + (m - m')C,$$
  $$e(a'P_{pub}, Q_{ID}) = e(aP_{pub}, Q_{ID})e(C, S_{ID})^{m-m'}.$$

  Note that

  $$\mathrm{Hash}(I, m', a'P, e(a'P_{pub}, Q_{ID})) = \mathrm{Hash}(I, m, aP, e(aP_{pub}, Q_{ID}))$$

and

$$\begin{aligned} e(a'P_{pub}, Q_{ID}) &= e(a'P, S_{ID}) \\ &= e(aP + (m - m')C, S_{ID}) \\ &= e(aP, S_{ID})e(C, S_{ID})^{m-m'} \\ &= e(aP_{pub}, Q_{ID})e(C, S_{ID})^{m-m'} \end{aligned}$$

Besides, for any collision $(a'P, e(a'P_{pub}, Q_{ID}))$, the holder of the trapdoor key S_{ID} can convince any third party that $< e(P, P), e(P, S_{ID}), e(a'P, P), e(a'P, S_{ID}) >$ is a valid Diffie-Hellman tuple, using the identity-based proof of knowledge for the equality of two discrete logarithms. In particular, it also holds for the original input $(aP, e(aP_{pub}, Q_{ID}))$.

Therefore, the forgery is successful. Moreover, if $(aP, e(aP_{pub}, Q_{ID}))$ is uniformly distributed, then the distribution of $(a'P, e(a'P_{pub}, Q_{ID}))$ is computationally indistinguishable from uniform.

Theorem 7. *The construction above is a secure identity-based chameleon hash scheme under the assumption that the BDHP in* $(\mathbb{G}_1, \mathbb{G}_2, e)$ *is intractable.*

Proof. We prove that the construction above satisfies the properties defined in section 3.1.

- **Collision resistance:** Assume to the contrary, that there exists a polynomial time algorithm $\mathcal{A}$, with a non-negligible probability, that outputs two pairs $(m, aP, e(aP_{pub}, Q_{ID}))$ and $(m', a'P, e(a'P_{pub}, Q_{ID}))$ which satisfy $\mathrm{Hash}(I, m', a'P, e(a'P_{pub}, Q_{ID})) = \mathrm{Hash}(I, m, aP, e(aP_{pub}, Q_{ID}))$, we can compute $e(C, S_{ID}) = (e(a'P_{pub}, Q_{ID})/e(aP_{pub}, Q_{ID}))^{(m-m')^{-1}}$ efficiently. This is equivalent to solve the BDHP in $\mathbb{G}_1$.
- **Semantic security:** Given a value $\mathcal{H}$, a customized identity I, and any message m, there exists exactly one pair $(aP, e(aP_{pub}, Q_{ID}))$ such that $\mathcal{H} = \mathrm{Hash}(I, m, aP, e(aP_{pub}, Q_{ID}))$.
- **Key exposure freeness:** Note that even if the adversary has obtained polynomially many signatures $e(C_j, S_{ID})$ on message I_j, he can not forge a signature $e(C, S_{ID})$ on message $I \neq I_j$, otherwise the Libert and Quisquater's identity-based undeniable signature scheme [22] can be broken. However, the unforgeability of this scheme with non-interactive zero-knowledge proof confirmation and disavowal protocols is proved to be equivalent to the BDHP. Therefore, even if the adversary has oracle access to $\mathcal{F}$ and is allowed polynomially many queries on triples $(I_j, m_j, a_jP, e(a_jP_{pub}, Q_{ID}))$ of his choice, there is no efficient algorithm for him to find a collision of the hash value $\mathcal{H} = \mathrm{Hash}(I, m, aP, e(aP_{pub}, Q_{ID}))$ where $I \neq I_j$. □

Similarly, we can construct an identity-based chameleon signature scheme by incorporating the proposed identity-based chameleon hash scheme and any secure identity-based signature scheme.

7 Conclusions

Chameleon signatures simultaneously provide the properties of non-repudiation and non-transferability for the signed message, thus can be used to solve the conflict between authenticity and privacy in the digital signatures. However, the original constructions suffer from the so-called key exposure problem of chameleon hashing. Recently, some specific constructions of key-exposure free chameleon hashing and signatures are presented, based on the RSA assumption or bilinear pairings.

In this paper, we propose the first key-exposure free chameleon hash scheme based on discrete logarithm systems, without using the gap Diffile-Hellman groups. Besides, one distinguished advantage of the resulting chameleon signature scheme is that the property of "message hiding" or "message recovery" can be achieved freely by the signer. Moreover, we propose the first identity-based chameleon hash scheme without key exposure, which gives a positive answer for the open problem introduced by Ateniese and de Mederious in 2004.

References

1. G. Ateniese and B. de Medeiros, *Identity-based chameleon hash and applications*, FC 2004, LNCS 3110, pp.164-180, Springer-Verlag, 2004.
2. G. Ateniese and B. de Medeiros, *On the key exposure problem in chameleon hashes*, SCN 2004, LNCS 3352, pp.165-179, Springer-Verlag, 2005.
3. D. Boyar, D. Chaum, and D. Damgård, *Convertible undeniable signatures*, Advances in Cryptology-Crypto 1990, LNCS 537, pp.183-195, Springer-Verlag, 1991.
4. P. Barreto, H. Kim, B. Lynn, and M. Scott, *Efficient algorithms for Pairing-based cryptosystems*, Advances in Cryptology-Crypto 2002, LNCS 2442, pp.354-368, Springer-Verlag, 2002.
5. D. Boneh, B. Lynn, and H. Shacham, *Short signatures from the Weil pairings*, Advances in Cryptology-Asiacrypt 2001, LNCS 2248, pp.514-532, Springer-Verlag, 2001.
6. D. Boneh and M. Franklin, *Identity-based encryption from the Weil pairing*, Advances in Cryptology-Crypto 2001, LNCS 2139, pp.213-229, Springer-Verlag, 2001.
7. J. Baek and Y. Zheng, *Identity-based threshold decryption*, PKC 2004, LNCS 2947, pp.248-261, Springer-Verlag, 2004.
8. M. Bellare and P. Rogaway, *The exact security of digital signatures-How to sign with RSA and Rabin*, Advances in Cryptology -Eurocrypt 1996, LNCS 1070, pp.399-416, Springer-Verlag, 1996.
9. J. Cha and J. Cheon, *An identity-based signature from gap Diffie-Hellman groups*, PKC 2003, LNCS 2567, pp.18-30, Springer-Verlag, 2003.
10. D. Chaum, *Designated confirmer signatures*, Advances in Cryptology-Eurocrypt 1994, LNCS 950, pp.86-91, Springer-Verlag, 1994.
11. D. Chaum, *Zero-knowledge undeniable signatures*, Advances in Cryptology-Eurocrypt 1990, LNCS 473, pp.458-464, Springer-Verlag, 1991.
12. D. Chaum and H. van Antwerpen, *Undeniable signatures*, Advances in Cryptology-Crypto 1989, LNCS 435, pp.212-216, Springer-Verlag, 1989.
13. J. Coron, *On the exact security of full domain hash*, Advances in Cryptology-Crypto 2000, LNCS 1880, pp.229-235, Springer-Verlag, 2000.
14. D. Chaum and T. Pedersen, *Wallet databases with observers*, Advances in Cryptology-Crypto 1992, LNCS 740, pp.89-105, Springer-Verlag, 1993.
15. X. Chen, F. Zhang, and K. Kim, *Chameleon hashing without key exposure*, ISC 2004, LNCS 3225, pp.87-98, Springer-Verlag, 2004.
16. S. Galbraith, W. Mao, and K. G. Paterson, *RSA-based undeniable signatures for general moduli*, CT-RSA 2002, LNCS 2271, pp.200-217, Springer-Verlag, 2002.
17. S. Galbraith and W. Mao, *Invisibility and anonymity of undeniable and confirmer signatures*, CT-RSA 2003, LNCS 2612, pp.80-97, Springer-Verlag, 2003.
18. S. Gennaro, H. Krawczyk, and T. Rabin, *RSA-based undeniable signatures*, Advances in Cryptology-Crypto 1997, LNCS 1294, pp.132-149, Springer-Verlag, 1997.
19. K. Kurosawa, and S. Heng, *3-move undeniable signature scheme*, Advances in Cryptology-Eurocrypt 2005, LNCS 3494, pp.181-197, Springer-Verlag, 2005.
20. H. Krawczyk and T. Rabin, *Chameleon hashing and signatures*, Proc. of NDSS 2000, pp.143-154, 2000.

21. F. Hess, *Efficient identity based signature schemes based on pairings*, SAC 2002, LNCS 2595, pp.310-324, Springer-Verlag, 2002.
22. B. Libert and J. Quisquater, *ID-based undeniable signatures*, CT-RSA 2004, LNCS 2694, pp.112-125, Springer-Verlag, 2004.
23. V. Miller, *The Weil pairing, and its efficient calculation*, Journal of Cryptology, 17(4), pp.235-261, Springer-Verlag, 2004.
24. W. Ogata, K. Kurosawa, and S. Heng, *The security of the FDH variant of Chaum's undeniable signature scheme*, PKC 2005, LNCS 3386, pp.328-345, Springer-Verlag, 2005.
25. T. Okamoto and D. Pointcheval, *The gap-problems: a new class of problems for the security of cryptographic Schemes*, PKC 2001, LNCS 1992, pp.104-118, Springer-Verlag, 2001.
26. C. P. Schnorr, *Efficient signature generation for smart cards*, Journal of Cryptology, 4(3), pp.239-252, Springer-Verlag, 1991.
27. A. Shamir, *Identity-based cryptosystems and signature schemes*, Advances in Cryptology-Crypto 1984, LNCS 196, pp.47-53, Springer-Verlag, 1984.
28. A. Shamir and Y. Tauman, *Improved online/offline signature schemes*, Advances in Cryptology-Crypto 2001, LNCS 2139, pp.355-367, Springer-Verlag, 2001.