Information Sciences 178 (2008) 4192–4203

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Efficient generic on-line/off-line (threshold) signatures without key exposure ☆

Xiaofeng Chen [a,*], Fangguo Zhang [a], Haibo Tian [a], Baodian Wei [a], Willy Susilo [b], Yi Mu [b], Hyunrok Lee [c], Kwangjo Kim [c]

[a] *School of Information Science and Technology, Sun Yat-Sen University, Guangzhou 510275, PR China*
[b] *Centre for Computer and Information Security Research, School of Computer Science and Software Engineering, University of Wollongong, Australia*
[c] *International Research Center for Information Security (IRIS), Information and Communications University, Taejon 305-714, Republic of Korea*

ARTICLE INFO

Article history:
Received 18 December 2007
Received in revised form 28 April 2008
Accepted 24 June 2008

Keywords:
On-line/off-line
Threshold signatures
Double-trapdoor
Chameleon hashing
Key exposure problem

ABSTRACT

The "hash–sign–switch" paradigm was firstly proposed by Shamir and Tauman with the aim to design an efficient on-line/off-line signature scheme. Nonetheless, all existing on-line/off-line signature schemes based on this paradigm suffer from the key exposure problem of chameleon hashing. To avoid this problem, the signer should pre-compute and store a plenty of different chameleon hash values and the corresponding signatures on the hash values in the off-line phase, and send the collision and the signature for a certain hash value in the on-line phase. Hence, the computation and storage cost for the off-line phase and the communication cost for t0he on-line phase in Shamir–Tauman's signature scheme are still a little more overload. In this paper, we first introduce a special double-trapdoor hash family based on the discrete logarithm assumption and then incorporate it to construct a more efficient generic on-line/off-line signature scheme without key exposure. Furthermore, we also present the first key-exposure-free generic on-line/off-line threshold signature scheme without a trusted dealer. Additionally, we prove that the proposed schemes have achieved the desired security requirements.

1. Introduction

The notion of on-line/off-line signatures was introduced by Even et al. [12,13]. The idea is to perform the signature generating procedure in two phases. The first phase is performed off-line (before the message to be signed is known) and the second phase is performed on-line (after the message to be signed is known). On-line/off-line signatures are particularly useful in smart card applications [29]: The off-line phase is implemented either during the card manufacturing process or as a background computation whenever the card is connected to power, and the on-line phase uses the stored result of the off-line phase to sign actual messages. The on-line phase is typically very fast, and hence can be extended efficiently even on a weak processor.

Even et al. proposed a general method for converting any signature scheme into an on-line/off-line signature scheme. Nonetheless, the method is rather impractical since it increases the size of the signature by a quadratic factor. In Crypto 2001, Shamir and Tauman [29] used the so-called "chameleon hash functions" to develop a new paradigm, named "hash–sign–switch", for designing efficient on-line/off-line signature schemes.

☆ Parts of this paper appeared in the 5th International Conference on Applied Cryptography and Network Security, LNCS 4521, pp. 18–30, Springer-Verlag, 2007.

* Corresponding author.
E-mail address: isschxf@mail.sysu.edu.cn (X. Chen).

Chameleon hash functions, firstly introduced by Krawczyk and Rabin [21], are trapdoor one-way hash functions which prevent anyone, except the holder of the trapdoor information from computing the collisions for every given input. Chameleon hash functions were originally used to design chameleon signatures, which as well as undeniable signatures [6] simultaneously provide non-repudiation and non-transferability for the signed message. In the chameleon signature schemes, the recipient is the holder of trapdoor information, while in the case of on-line/off-line signatures, the signer is the holder of the trapdoor information. Therefore, in the off-line phase the signer generates a signature σ by using a provably secure signature scheme to sign the chameleon hash value $h(m', r')$ of a random message m' and a random auxiliary number r'. In the on-line phase, the signer computes a collision r of the chameleon hash function for the given message m such that $h(m, r) = h(m', r')$. The signature for the message m is the pair (σ, r).

In the Shamir–Tauman's on-line/off-line signature schemes, one limitation is that the signatures for different messages must use different chameleon hash values. Otherwise, if the signer uses the same hash value twice to obtain two signatures on two different messages, then the recipient can obtain a hash collision and use it to recover the signer's trapdoor information, which is the signer's secret key. This problem is known as the *key exposure* problem of chameleon hashing. To avoid this problem, the signer must compute and store a large number of different chameleon hash values and the corresponding signatures on the hash values in the off-line phase. Given a message in the on-line phase, the signer first selects a one-time hash value, and then computes a hash collision for the hash value. Then, he provides the hash collision and the corresponding signature to the verifier. Hence, the computation and storage cost for the off-line phase and the communication cost for the on-line phase in Shamir–Tauman's signature scheme are still a little more overload.

The idea of threshold cryptography [10] is to distribute a secret information (*i.e.*, a secret key) and computation (*i.e.*, decryption or signature generation) among multi parties in order to prevent a single point of failure or abuse. In a threshold signature scheme, given a group of n players, and a threshold $t < n$, no subset of the players of size at most t can generate a signature. Recently, Crutchfield et al. [7] presented a generic on-line/off-line threshold signature scheme without trusted dealers. The main idea of this work is to apply the "hash–sign–switch" paradigm to a threshold signature scheme. Therefore, the resulting scheme will suffer from the inherent key exposure problem.

Our contributions. In this paper, we first address the key exposure problem by introducing a double-trapdoor hash family based on the discrete logarithm assumption. Then, we apply the "hash–sign–switch" paradigm to propose a more efficient generic on-line/off-line signature scheme. The key idea is as follows: the hash value and the corresponding signature are always identical and can be viewed as the public key of the signer. Hence, it is not required to compute and store them in the off-line phase. Additionally, we introduce the idea of long-term trapdoor and one-time trapdoor in our chameleon hash families, which is similar to the idea of master trapdoor and specific trapdoor in the multi-trapdoor commitment schemes [16]. The one-time trapdoor is used only once for each message being signed in the on-line phase, which prevents the recipient from recovering the trapdoor information of the signer and computing other collisions. Furthermore, we also propose the first key-exposure-free on-line/off-line threshold signature scheme without trusted dealers. We prove that the proposed signature schemes can achieve the desired security requirements in the random oracle model.

In order to achieve the communication and computation advantages of our on-line/off-line signature scheme, we adopt elliptic curve cryptosystems [20,24] to present our double-trapdoor hash family. Certainly, we can design such a double-trapdoor hash family over other generic groups, *e.g.*, the subgroup of $\mathbb{Z}_p^*$. However, we argue that such a double-trapdoor hash family over other generic groups is unsuitable for designing efficient generic on-line/off-line signature schemes. The reason is as follows: Since the "hash-sign-switch" paradigm is a generic method, it is required that any provably secure signature scheme $\mathscr{S}$ can be used to design the on-line/off-line signature scheme. However, only when the signature length of original signature scheme $\mathscr{S}$ is less than that of a group element, our proposed on-line/off-line signature scheme is superior to Shamir–Tauman's scheme in communication cost.[1] Currently, for any provably secure signature scheme, the signature length is more than 160 bits. Therefore, the elliptic curve cryptosystems seem to be the optimal choice. If we adopt other generic group such as the subgroup of $\mathbb{Z}_p^*$, some short signature schemes [3,5,28,30] can not be used to design our on-line/off-line signature scheme. For more details, we refer the readers to Section 4.3.

1.1. Related work

As noted in [29], some signature schemes such as Fiat–Shamir, Schnorr, and ElGamal signature schemes [14,27,11] can be naturally partitioned into on-line and off-line phases. The reason is that the first step in these signature schemes does not depend on the given message, and can thus be carried out off-line. However, these are particular schemes with special structure and specific security assumptions rather than a general and provably secure conversion technique for arbitrary signature schemes. Shamir and Tauman introduced the "hash-sign-switch" method for simultaneously improving both the security and the real-time efficiency of any signature scheme by converting it into an efficient on-line/off-line signature scheme. Generally, a new chameleon hash family results in a new on-line/off-line signature scheme. Recently, some variants of on-line/off-line signature schemes [4,7,22] have been proposed based on Shamir–Tauman's general construction.

The key exposure problem of chameleon hashing is firstly addressed by Ateniese and de Medeiros [1] in the original chameleon signature schemes. Chen et al. [8] proposed the first full construction of a chameleon hash function without key

[1] In any case, our proposed scheme is no inferior to Shamir–Tauman's scheme in computation and storage cost.

exposure. Later, Ateniese and de Medeiros presented several constructions of exposure-free chameleon hash functions based on different cryptographic assumptions [2]. However, it seems that the current chameleon hash schemes without key exposure are not suitable for designing efficient on-line/off-line signature schemes. The reasons are twofold: Firstly, collision computation in these chameleon hash schemes usually requires the costly modular exponentiation operation. Secondly, though collision forgery will not reveal the signer's trapdoor information, it allows the verifier to compute other collisions for the same hash value. To the best of our knowledge, there exists no work that solves the key exposure problem in the generic on-line/off-line signature schemes except [9].

1.2. Organization of the paper

The rest of the paper is organized as follows: some preliminaries are provided in Section 2. The new double-trapdoor chameleon hash family based on the discrete logarithm assumption is presented in Section 3. Our efficient generic on-line/off-line signature scheme and its security and efficiency analysis are given in Section 4. The resulting on-line/off-line threshold signature scheme is given in Section 5. Finally, Section 6 concludes the paper.

2. Preliminaries

In this section, we firstly introduce the notion of chameleon hash family and the "hash–sign–switch" paradigm [29]. We then present the definition of secure distributed key generation protocol and threshold signature scheme [17].

2.1. Chameleon hash family

Definition 1 (*Chameleon hash family*). A chameleon hash family consists of a pair $(\mathscr{I}, \mathscr{H})$:

- $\mathscr{I}$ is a probabilistic polynomial-time key generation algorithm that on input 1^k, outputs a pair (HK, TK) such that the sizes of HK, TK are polynomially related to k.
- $\mathscr{H}$ is a family of randomized hash functions. Every hash function in $\mathscr{H}$ is associated with a hash key HK, and is applied to a message from a space $\mathscr{M}$ and a random element from a finite space $\mathscr{R}$. The output of the hash function H_{HK} does not depend on TK.

A chameleon hash family $(\mathscr{I}, \mathscr{H})$ has the following properties:

(1) *Efficiency:* Given a hash key HK and a pair $(m, r) \in \mathscr{M} \times \mathscr{R}$, $H_{HK}(m, r)$ is computable in polynomial time.
(2) *Collision resistance:* There is no probabilistic polynomial time algorithm $\mathscr{A}$ that on input HK outputs, with a probability which is not negligible, two pairs $(m_1, r_1), (m_2, r_2) \in \mathscr{M} \times \mathscr{R}$ that satisfy $m_1 \neq m_2$ and $H_{HK}(m_1, r_1) = H_{HK}(m_2, r_2)$ (the probability is over HK, where $(HK, TK) \leftarrow \mathscr{I}(1^k)$, and over the random coin tosses of algorithm $\mathscr{A}$).
(3) *Trapdoor collisions:* There exists a probabilistic polynomial time algorithm that given a pair $(HK, TK) \leftarrow \mathscr{I}(1^k)$, a pair $(m_1, r_1) \in \mathscr{M} \times \mathscr{R}$, and an additional message $m_2 \in \mathscr{M}$ that satisfy $m_1 \neq m_2$, outputs a value $r_2 \in \mathscr{R}$ such that:
   - $H_{HK}(m_1, r_1) = H_{HK}(m_2, r_2)$.
   - If r_1 is uniformly distributed in $\mathscr{R}$ then the distribution of r_2 is computationally indistinguishable from uniform in $\mathscr{R}$.

2.2. Shamir–Tauman's "hash–sign–switch" paradigm

Shamir and Tauman introduced the following"hash–sign–switch" paradigm to construct a generic on-line/off-line signature scheme.

- **System parameters generation:** Let $(\mathscr{I}, \mathscr{H})$ be any trapdoor hash family and $(\mathscr{G}, \mathscr{S}, \mathscr{V})$ be any provably secure signature scheme. The system parameters are $SP = \{(\mathscr{I}, \mathscr{H}), (\mathscr{G}, \mathscr{S}, \mathscr{V})\}$.
- **Key generation algorithm:**
  – On input 1^k, run the key generation algorithm of the original signature scheme $\mathscr{G}$ to obtain a signing/verification key pair (SK, VK).
  – On input 1^k, run the key generation algorithm of the trapdoor hash family $(\mathscr{I}, \mathscr{H})$ to obtain a hash/trapdoor key pair (HK, TK).

  The signing key is (SK, TK) and the verification key is (VK, HK).
- **The signing algorithm:**

1. Off-line phase
   – Choose at random $(m_i, r_i) \in_R \mathscr{M} \times \mathscr{R}$, and compute the chameleon hash value $h_i = H_{HK}(m_i, r_i)$.
   – Run the signing algorithm $\mathscr{S}$ with the signing key SK to sign the message h_i. Let the output be $\sigma_i = \mathscr{S}_{SK}(h_i)$.
   – Store the pair (m_i, r_i), and the signature σ_i.

2. On-line phase
   - For a given message m, retrieve from the memory a random pair (m_i, r_i) and the signature σ_i.
   - Compute $r \in \mathscr{R}$ such that $H_{HK}(m, r) = H_{HK}(m_i, r_i)$.
   - Send (r, σ_i) as the signature of the message m.

- **The verification algorithm:**
  - Compute $h_i = H_{HK}(m, r)$.
  - Verify that σ_i is indeed a signature of the hash value h_i with respect to the verification key VK.

In the following, we present Shamir–Tauman's "hash–sign–switch" paradigm with elliptic curve analogue of the chameleon hash family based on the discrete logarithm assumption [21,29], so that we can fairly compare it with our proposed signature scheme.

- **System parameters generation:** Let l be a prime power, and $E(\mathbb{F}_l)$ an elliptic curve over finite field $\mathbb{F}_l$. Let $\#E(\mathbb{F}_l)$ be the number of points of $E(\mathbb{F}_l)$, and P be a point of $E(\mathbb{F}_l)$ with prime order q where $q|\#E(\mathbb{F}_l)$. Denote $\mathbb{G}$ the subgroup generated by P. Let $(\mathscr{I}, \mathscr{H})$ be the trapdoor hash family based on the discrete logarithm assumption and $(\mathscr{G}, \mathscr{S}, \mathscr{V})$ be any provably secure signature scheme. The system parameters are $SP = \{E, l, q, P, \mathbb{G}, (\mathscr{G}, \mathscr{S}, \mathscr{V})\}$.
- **Key generation algorithm:**
  - On input 1^k, run the key generation algorithm of the original signature scheme $\mathscr{G}$ to obtain the signing/verification key pair (SK, VK).
  - On input 1^k, run the key generation algorithm of the trapdoor hash family $(\mathscr{I}, \mathscr{H})$ to obtain the hash/trapdoor key pair $(Y = xP, x)$.

  The signing key is (SK, x) and the verification key is (VK, Y).[2]
- **The signing algorithm:**

1. Off-line phase
   - Choose at random $(m_i, r_i) \in_R \mathscr{M} \times \mathscr{R}$, and compute the chameleon hash value $h_i = H_Y(m_i, r_i) = m_iP + r_iY$.
   - Run the signing algorithm $\mathscr{S}$ with the signing key SK to sign the message h_i. Let the output be $\sigma_i = \mathscr{S}_{SK}(h_i)$.
   - Store the pair (m_i, r_i), and the signature σ_i.

2. On-line phase
   - For a given message m, retrieve from the memory x^{-1} and a random pair (m_i, r_i).
   - Compute $r = x^{-1}(m_i - m) + r_i \bmod q$.
   - Send (r, σ_i) as the signature of the message m.

- **The verification algorithm:**
  - Compute $h_i = H_Y(m, r) = mP + rY$.
  - Verify that σ_i is indeed a signature of the hash value h_i with respect to the verification key VK.

2.3. Secure distributed key generation protocol

A distributed key generation (DKG) protocol is an essential component of threshold cryptosystems in order to construct the public key and secret key. In a DKG protocol with n players, the public key is made known to all players, whereas the secret key is known by none. Instead, each player receives a key share, from which they can recover the secret key.

Feldman [15] presented a verifiable secret sharing (VSS) protocol, denoted by Feldman-VSS, which allows a *trusted* dealer to share a key x among n parties that is secure with threshold t. Pederson [25] proposed the first DKG protocol based on Feldman-VSS. However, Gennaro et al. [18] showed a security flaw of Pederson's DKG protocol and then proposed a secure DKG protocol for discrete logarithm cryptosystems. After performing the DKG protocol, each player obtains a secret share of a random, uniformly distributed value x and the secret/public key pair is $(x, y = g^x)$. Also, this protocol has the property of "simulatability", *i.e.*, for any adversary $\mathscr{A}$ who corrupts a set of players $\mathscr{P}_{\mathscr{A}} \in \mathscr{P}$, there exists a simulator $\mathscr{M}$, such that on input an element y, can produce an output distribution which is polynomially indistinguishable from $\mathscr{A}$'s view of a run of the DKG protocol that ends with y as the public key output.

We will use this DKG protocol throughout this paper.

[2] The value of x^{-1} should be pre-computed and stored in order to decrease the computation cost in the on-line phase of the signature scheme.

2.4. Threshold signature scheme

Gennaro et al. presented a formal definition for threshold signatures [17]. Given a signature scheme $\mathscr{S} = (\mathsf{Key-Gen}, \mathsf{Sig}, \mathsf{Ver})$, a (t, n) threshold signature scheme $\mathscr{TS}$ for $\mathscr{S}$ is a triple of randomized algorithms (Thresh-Key-Gen, Thresh-Sig, Ver) for a set of n players $\mathscr{P} = \{p_1, p_2, \ldots, p_n\}$ with a threshold value t where:

- Thresh-Key-Gen is a distributed key generation algorithm used by the players to create a public/secret key pair $(PK, SK) \in \mathscr{PK} \times \mathscr{SK}$ such that each $p_i \in \mathscr{P}$ receives a share SK_i of the secret key SK.
- Thresh-Sig is a distributed signing algorithm used by the players to create a signature for a message $m \in \mathscr{M}$ such that the output of the algorithm is $\mathscr{S}_{SK}(m)$. This algorithm can be decomposed into two algorithms: signature share generation and signature reconstruction.
- Ver is a verification algorithm such that $\mathsf{Ver}(PK, m, \sigma) = 1$ if and only if $\sigma = \mathscr{S}_{SK}(m)$.

A (t, n) threshold signature scheme $\mathscr{TS}$ is said to be unforgeable, if no malicious adversary who corrupts at most t players can produce the signature on any new message $m \neq m_i$, given the view of the protocol Thresh-Key-Gen and of the protocol Thresh-Sig on input the adversary's adaptively chosen messages m_i for $1 \leqslant i \leqslant k$.

3. A new double-trapdoor chameleon hash family

In this section, we first provide a formal definition of double-trapdoor chameleon hash family. We then propose a new construction based on the discrete logarithm assumption.

3.1. Formal definition

Definition 2 (*Double-trapdoor chameleon hash family*). A double-trapdoor chameleon hash family consists of a triple $(\mathscr{I}_1, \mathscr{I}_2, \mathscr{H})$:

- $\mathscr{I}_1$ is a probabilistic polynomial-time key generation algorithm that on input 1^k, outputs a *long-term* hash/trapdoor key pair (HK_1, TK_1) such that the sizes of HK_1, TK_1 are polynomially related to k. Note that (HK_1, TK_1) is associated with all chameleon hash functions in the family and can be used repeatedly during its life span.
- $\mathscr{I}_2$ is a probabilistic polynomial-time key generation algorithm that on input 1^k, outputs a *one-time* hash/trapdoor key pair (HK_2, TK_2) such that the sizes of HK_2, TK_2 are polynomially related to k. Note that (HK_2, TK_2) is only associated with a specific chameleon hash function in the family and can be used only once.
- $\mathscr{H}$ is a family of randomized hash functions. Every hash function in $\mathscr{H}$ is associated with a hash key pair $HK = (HK_1, HK_2)$, and is applied to a message from a space $\mathscr{M}$ and a random element from a finite space $\mathscr{R}$. The output of the hash function H_{HK} does not depend on $TK = (TK_1, TK_2)$.

A double-trapdoor chameleon hash family $(\mathscr{I}_1, \mathscr{I}_2, \mathscr{H})$ has the following properties:

(1) *Efficiency:* Given a hash key pair HK and a pair $(m, r) \in \mathscr{M} \times \mathscr{R}$, H_{HK} is computable in polynomial time.
(2) *Collision resistance:* There is no probabilistic polynomial time algorithm $\mathscr{A}$ that on input HK outputs, with a probability which is not negligible, two pairs $(m_1, r_1), (m_2, r_2) \in \mathscr{M} \times \mathscr{R}$ that satisfy $m_1 \neq m_2$ and $H_{HK}(m_1, r_1) = H_{HK}(m_2, r_2)$ (the probability is over HK, where $(HK_1, TK_1) \leftarrow \mathscr{I}_1(1^k)$, $(HK_2, TK_2) \leftarrow \mathscr{I}_2(1^k)$ and over the random coin tosses of algorithm $\mathscr{A}$).
(3) *Trapdoor collisions:* There exists a probabilistic polynomial time algorithm that given a pair $(HK_1, TK_1) \leftarrow \mathscr{I}_1(1^k)$, a pair $(HK_2, TK_2) \leftarrow \mathscr{I}_2(1^k)$, a pair $(m_1, r_1) \in \mathscr{M} \times \mathscr{R}$, and an additional message $m_2 \in \mathscr{M}$ that satisfy $m_1 \neq m_2$, outputs a value $r_2 \in \mathscr{R}$ such that:
    - $H_{HK}(m_1, r_1) = H_{HK}(m_2, r_2)$.
    - If r_1 is uniformly distributed in $\mathscr{R}$ then the distribution of r_2 is computationally indistinguishable from uniform in $\mathscr{R}$.
(4) *Key-exposure freeness:* There is no probabilistic polynomial time algorithm $\mathscr{A}$ that on input a long-term hash key HK_1, two different one-time hash key HK_2 and HK'_2, two pairs $(m_1, r_1), (m_2, r_2) \in \mathscr{M} \times \mathscr{R}$ that satisfy $m_1 \neq m_2$ and $H_{HK}(m_1, r_1) = H_{HK'}(m_2, r_2)$ outputs, with a probability which is not negligible, the long-term trapdoor key TK_1.

3.2. A new construction

- **System parameters generation:** Let l be a prime power, and $E(\mathbb{F}_l)$ an elliptic curve over finite field $\mathbb{F}_l$. Let $\#E(\mathbb{F}_l)$ be the number of points of $E(\mathbb{F}_l)$, and P be a point of $E(\mathbb{F}_l)$ with prime order q where $q | \#E(\mathbb{F}_l)$. Denote by $\mathbb{G}$ the subgroup generated by P. Define a collision resistant hash function $f : \mathbb{Z}_q \times \mathbb{G} \rightarrow \mathbb{Z}_q$. Choose two random elements $k, x \in_R \mathbb{Z}_q^*$, and compute $K = kP, Y = xP$. The one-time/long-term hash key pair is $HK = (K, Y)$, and the one-time/long-term trapdoor key pair is $TK = (k, x)$.

- **The hash family:** Given the hash key pair HK, the proposed chameleon hash function $H_{HK} : \mathbb{Z}_q \times \mathbb{Z}_q \to \mathbb{G}$ is defined as follows:

$$H_{HK}(m, r) = f(m, K) \cdot (K + Y) + rP.$$

Theorem 1. *The construction above is a double-trapdoor chameleon hash family under the assumption that the discrete logarithm problem in $\mathbb{G}$ is intractable.*

Proof. We prove that the scheme satisfies the properties defined in Section 3.1.

(1) *Efficiency:* Given the hash key pair (K, Y) and a pair $(m, r) \in \mathbb{Z}_q \times \mathbb{Z}_q$, $H_{HK}(m, r) = f(m, K) \cdot (K + Y) + rP$ is computable in polynomial time.

(2) *Collision resistance:* Assume to the contrary, that there exists a polynomial time algorithm $\mathscr{A}$ that on input HK outputs, with a probability which is not negligible, two pairs $(m_1, r_1), (m_2, r_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$ that satisfy $m_1 \neq m_2$ and $H_{HK}(m_1, r_1) = H_{HK}(m_2, r_2)$. Then, we can use $\mathscr{A}$ to solve the discrete logarithm problem in $\mathbb{G}$ as follows: For a randomly given instance (P, aP), choose a random integer $b \in_R \mathbb{Z}_q$ and define $K = aP$, and $Y = bP$. Therefore, if the equation $f(m_1, aP) \cdot (aP + bP) + r_1P = f(m_2, aP) \cdot (aP + bP) + r_2P$ holds, we can compute

$$a = (f(m_1, aP) - f(m_2, aP))^{-1}(r_2 - r_1) - b \bmod q.$$

(3) *Trapdoor collisions:* Assume that we are given the hash key pair (K, Y), the trapdoor key pair (k, x), a pair $(m_1, r_1) \in \mathbb{Z}_q \times \mathbb{Z}_q$, and an additional message $m_2 \in \mathbb{Z}_q$, we want to find $r_2 \in \mathbb{Z}_q$ such that

$$f(m_1, K) \cdot (K + Y) + r_1P = f(m_2, K) \cdot (K + Y) + r_2P.$$

The value of r_2 can be computed in polynomial time as follows:

$$r_2 = r_1 + (k + x)(f(m_1, K) - f(m_2, K)) \bmod q.$$

Also, if r_1 is uniformly distributed in $\mathscr{R}$ then the distribution of r_2 is computationally indistinguishable from uniform in $\mathscr{R}$.

(4) *Key-exposure freeness:* Given the long-term hash key Y, two different one-time hash key (K, K'), two pairs $(m_1, r_1), (m_2, r_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$ that satisfy $m_1 \neq m_2$ and $f(m_1, K) \cdot (K + Y) + r_1P = f(m_2, K') \cdot (K' + Y) + r_2P$, *i.e.*, $f(m_1, K) \cdot (k + x) + r_1 = f(m_2, K') \cdot (k' + x) + r_2 \bmod q$, it is infeasible to compute x since there are another two unknown integers k and k' which are independent of x in this equation. Moreover, it is equivalent to solve the discrete logarithm problem in $\mathbb{G}$ to compute k (or k') from K (or K'). □

Remark 1. Note that the hash key K must be used only once. Otherwise, given a collision (m_1, r_1) and (m_2, r_2) that satisfy $m_1 \neq m_2$, we can compute $k + x$ from the equation $f(m_1, K) \cdot (K + Y) + r_1P = f(m_2, K) \cdot (K + Y) + r_2P$. Though the trapdoor information x is still not revealed because k is unknown, it can be used to compute other collisions for the same hash value. Note that this feature has some advantages in the chameleon signatures. For example, the signer can provide a different collision to hide the original signed message. While in the case of on-line/off-line signatures, it means that the verifier can universally forge a signature of the signer. This is the reason why we introduce the idea of one-time hash key in double-trapdoor chameleon hash family.

4. Our efficient on-line/off-line signature scheme

In this section, we apply the "hash–sign–switch" paradigm to propose a more efficient on-line/off-line signature scheme. We can adopt any provably secure digital signature scheme to design our on-line/off-line signature scheme, so it is also a generic construction.

4.1. The proposed signature scheme

The proposed on-line/off-line signature scheme consists of the following efficient algorithms:

- **System parameters generation:** Let l be a prime power, and $E(\mathbb{F}_l)$ an elliptic curve over finite field $\mathbb{F}_l$. Let $\#E(\mathbb{F}_l)$ be the number of points of $E(\mathbb{F}_l)$, and P be a point of $E(\mathbb{F}_l)$ with prime order q where $q | \#E(\mathbb{F}_l)$. Denote $\mathbb{G}$ the subgroup generated by P. Define a collision resistant hash function $f : \mathbb{Z}_q \times \mathbb{G} \to \mathbb{Z}_q$. Given a one-time/long-term hash key $HK = (K, Y)$, the chameleon hash function $H_{HK} : \mathbb{Z}_q \times \mathbb{Z}_q \to \mathbb{G}$ is defined as follows:

$$H_{HK}(m, r) = f(m, K) \cdot (K + Y) + rP.$$

Let $(\mathscr{G}, \mathscr{S}, \mathscr{V})$ be any provably secure signature scheme. The system parameters are $SP = \{E, l, q, P, \mathbb{G}, f, H_{HK}, (\mathscr{G}, \mathscr{S}, \mathscr{V})\}$.

- **Key generation algorithm:**
  - On input 1^k, run the key generation algorithm of the original signature scheme $\mathscr{G}$ to obtain the signing/verification key pair (SK, VK).
  - On input 1^k, run the key generation algorithm of the trapdoor hash family to obtain the *long-term* trapdoor/hash key pair $(x, Y = xP)$.
  - Choose at random $k^* \in_R \mathbb{Z}_q$, and compute the chameleon hash value $h = k^*P$. Run the signing algorithm $\mathscr{S}$ with the signing key SK to sign the message h. Let the output be $\sigma = S_{SK}(h)$.

  The signing key is (SK, x, k^*) and the verification key is (VK, Y, h, σ).
- **The signing algorithm:**

1. Off-line phase
   - Choose at random $k_i \in_R \mathbb{Z}_q$, and compute k_iP.
   - Store the *one-time* trapdoor/hash key pair (k_i, k_iP).

2. On-line phase
   - For a given message m, retrieve from the memory a random pair (k_i, k_iP).
   - Compute $r_i = k^* - f(m, k_iP)(k_i + x) \bmod q$.
   - Send (r_i, k_iP) as the signature of the message m.

- **The verification algorithm:**
  - Compute $h = f(m, k_iP)(k_iP + Y) + r_iP$ by using the *one-time* hash key k_iP and the *long-term* hash key Y.
  - Verify that σ is indeed a signature of the hash value h with respect to the verification key VK.[3]

Note that

$$h = f(m, k_iP)(k_iP + Y) + r_iP = f(m, k_iP)(k_iP + Y) + (k^* - f(m, k_iP)(k_i + x))P = k^*P$$

So, the proposed scheme satisfies the property of completeness.

Remark 2. Note that the value of x^{-1} need not be pre-computed and stored in our scheme, which is an improvement compared with [9,29]. Meanwhile, since $r_i = k^* - f(m, k_iP)(k_i + x) \bmod q$, it also requires only 1 modular multiplication of $\mathbb{Z}_q$ in the on-line phase of our scheme.

4.2. Security analysis of the proposed scheme

The most general known security requirement of a signature scheme is security against existential forgery on adaptively chosen message attacks, which was firstly defined by Goldwasser et al. [19] as follows:

Definition 3. A signature scheme $\Omega = (Gen, Sign, Ver)$ is existentially unforgeable under adaptive chosen message attacks if for any probabilistic polynomial time adversary $\mathscr{A}$ there exist no non-negligible probability ϵ such that

$$\mathbf{Adv}(\mathscr{A}) = \Pr\left[\begin{array}{l} \langle pk, sk\rangle \leftarrow \mathsf{Gen}(1^l); \\ \mathit{for}\ i = 1, 2, \ldots, k; \\ m_i \leftarrow \mathscr{A}(pk, m_1, \sigma_1, \ldots, m_{i-1}, \sigma_{i-1}), \sigma_i \leftarrow Sign(sk, m_i); \\ \langle m, \sigma\rangle \leftarrow \mathscr{A}(pk, m_1, \sigma_1, \ldots, m_k, \sigma_k); \\ m \notin \{m_1, \ldots, m_k\} \wedge Ver(pk, m, \sigma) = \text{accept} \end{array}\right] \geqslant \epsilon.$$

Now we give the formal security proof of our on-line/off-line signature scheme. More precisely, we have the following theorem:

Theorem 2. *In the random oracle model, the resulting on-line/off-line signature scheme is existentially unforgeable against adaptive chosen message attacks, provided that the discrete logarithm problem in $\mathbb{G}$ is intractable.*

Proof. In the proposed on-line/off-line signature scheme, the corresponding signature σ on the chameleon hash value h is viewed as the public key of the signer. Therefore, a hash collision r and a one-time hash key kP are the real signature on the message m.

[3] We argue that this step could be omitted since (h, σ) has been included in the verification key, *i.e.*, it is not required for the verifier to perform this step every time. Therefore, the verification algorithm of our signature scheme is much more efficient than that of [29].

Suppose that $\mathscr{A}$ is a probabilistic algorithm that given a verification key (VK, Y, h, σ), forges a signature with respect to the proposed on-line/off-line signature scheme by an adaptively chosen message attack in time T with success probability ϵ. We denote, respectively, by q_R and q_S the number of queries that $\mathscr{A}$ can at most ask to the random oracle and the signing oracle. Let (m_i, k_iP) denote the input of i-th query to the random oracle, and (r_i, k_iP) denote the corresponding signatures produced by the signing oracle. Let (m, r, kP) denote the output of $\mathscr{A}$. Since the success probability of $\mathscr{A}$ is ϵ, it follows that

$$Pr[\mathscr{V}_{VK}(h,\sigma) = 1 \wedge h = H_{kP,Y}(m,r) = H_{k_iP,Y}(m_i,r_i)] \geqslant \epsilon.$$

Then we can construct a probabilistic algorithm $\mathscr{M}$ to compute a for a randomly given instance (P, aP) where P is a generator of G as follows:

- Let (SK, VK) be the signing/verification key pair of the original signature scheme. Choose a random element $Y \in_R G$ and let Y be the long-term hash key. Define the chameleon hash value $h = aP$. Run the signing algorithm S with the signing key SK to sign the message h. Let the output be $\sigma = S_{SK}(h)$ and then publish the public key (VK, Y, h, σ).
- Maintain a list, called f-list, which is initially set to empty. If the ith query (m_i, K_i) to the hash oracle f is not in the list, choose a random element $e_i \in_R \mathbb{Z}_q$ and respond it as the answer of ith query. Then add (m_i, K_i, e_i) to the f-list.
- Let m_j denote the input of jth query to the signing oracle, choose at random $(e'_j, r'_j) \in_R \mathbb{Z}_q \times \mathbb{Z}_q$ (Note that e'_j is not in the f-list) and define

$$K'_j = e'^{-1}_j (h - r'_jP) - Y,$$

respond e'_j as the hash oracle answer to the query (m_j, K'_j), and (K'_j, r'_j) as the signing oracle answer to the query m_j. Then add (m_j, K'_j, e'_j) to the f-list.

Suppose the output of $\mathscr{A}$ is (m, K, r). If $m \neq m_j$ for $j = 1, \ldots, q_S$ and $h = f(m,K)(K + Y) + rP$, we say that $\mathscr{A}$ forges a signature (K, r) on the message m with respect to the proposed on-line/off-line signature scheme.

By replays of $\mathscr{A}$ with the same random tape but different choices of oracle f, as done in the forking Lemma [26], we can obtain two valid signatures (m, K, r) and (m, K, r') with respect to different hash oracles f and f'.

Note that $h = f(m,K)(K + Y) + rP$ and $h = f'(m,K)(K + Y) + r'P$, we can compute $a = (f'(m,K) - f(m,K))^{-1}(f'(m,K)r - f(m,K)r')$ as the discrete logarithm of aP with respect to the base P.

The success probability of $\mathscr{M}$ is also ϵ, and the running time of $\mathscr{M}$ is equal to the running time of the Forking Lemma which is bounded by $23Tq_R/\epsilon$ [26]. $\square$

4.3. Efficiency analysis of the proposed scheme

We compare the efficiency of our scheme with that of Shamir–Tauman's scheme given in Section 2.2 and Chen et al.'s scheme [9]. We denote by $C(\theta)$ the computation cost of operation θ, and by $|\lambda|$ the bits of λ. Also, we denote by M a scalar multiplication in G, by SM a simultaneous scalar multiplication of the form $\lambda P + \mu Q$ in G, by I the inversion in $\mathbb{Z}_q$, and by m the modular multiplication in $\mathbb{Z}_q$. We omit other operations such as point addition and hash in both schemes.

Since a 160-bit ECC key offers more or less the same level of security as a 1024-bit RSA key [23], we let $|q|$=160 in the following. Currently, for any secure signature scheme, the signature length $|\sigma| \geqslant |l| + 1$ since $|l|$ is about 160. In the optimal case, we can choose an elliptic curve $E(\mathbb{F}_l)$ such that $\#E(\mathbb{F}_l)$ is just a 160-bit prime q. From Hasse theorem, we know that $|l| = |\#E(\mathbb{F}_l)| = 160$.

Tables 1 and 2 present the comparison of the computation cost, the storage cost, and the communication cost for each message signing among Shamir–Tauman's scheme, Chen et al.'s scheme and our scheme.

Therefore, the proposed scheme is much superior to Shamir–Tauman's scheme in the computation cost of off-line phase, signature verification cost, storage cost and communication cost. The computation cost in the on-line phase is same. So, we argue that our signature scheme is more suitable for smart-card applications where both the computation and storage resources are limited. The disadvantage of our scheme is the higher computation cost in the key generation algorithm.

Compared with Chen et al.'s scheme [9], ours is more efficient in the key generation and off-line signing phase. Moreover, our signature scheme is more suitable for constructing an efficient generic on-line/off-line threshold signature scheme. For more details, refer to Section 5.

Table 1
Comparison of the computation cost

	Shamir–Tauman's scheme	Chen et al.'s scheme	Our scheme
Key generation	$1M + 1I$	$2M + 1I + 1C(\sigma)$	$2M + 1C(\sigma)$
Off-line phase	$1SM + 1C(\sigma)$	$1M + 1m$	$1M$
On-line phase	$1m$	$1m$	$1m$
Signature verification	$1SM + 1C(\sigma)$	$1SM$	$1SM$

Table 2
Comparison of the storage and communication cost

	Shamir–Tauman's scheme	Chen et al.'s scheme	Our scheme
Storage off-line phase	$2\|q\| + 1\|\sigma\|$	$2\|q\|$	$2\|q\|$
Communication on-line phase	$1\|q\| + 1\|\sigma\|$	$2\|q\|$	$2\|q\|$

Remark 3. We adopt elliptic curve cryptosystems [20,24] to present our double-trapdoor hash family as [9]. If we adopt other generic group such as the subgroup of $\mathbb{Z}_p^*$ to present our double-trapdoor chameleon hash family and on-line/off-line signature scheme, the communication cost for our on-line/off-line signature scheme is $1|q| + 1|p|$. For most current signature schemes, the signature length $|\sigma| < |p|$ if we let $|p| = 1024$. So, our proposed on-line/off-line signature scheme is inferior to Shamir–Tauman's scheme in communication cost since $1|q| + 1|p| > 1|q| + 1|\sigma|$. This is the reason why we choose the elliptic curve cryptosystems.

5. Generic on-line/off-line threshold signature scheme

Though Chen et al. [9] proposed the first generic on-line/off-line signature scheme without key exposure, we argue that it is not suitable for constructing an on-line/off-line threshold signature scheme by using Gennaro et al.'s DKG protocol [18]. It seems difficult to construct such an efficient DKG protocol without a trusted dealer: each player performs the protocol and obtains a secret share of a random, uniformly distributed value x^{-1} while the secret/public key pair is $(x, y = g^x)$.

In this section, we use our double-trapdoor chameleon hash family to present an efficient generic on-line/off-line threshold signature scheme without a trusted dealer $\mathscr{TS}^{\mathsf{On/Off}} = (\mathsf{Key\text{-}Gen}, \mathsf{Threshold\text{-}Sig\text{-}Off\text{-}line}, \mathsf{Threshold\text{-}Sig\text{-}On\text{-}line}, \mathsf{Verify})$. The proposed scheme uses Gennaro et al.'s DKG protocol and follows Crutchfield et al.'s construction [7].

5.1. The proposed on-line/off-line threshold signature scheme

The system parameters SP are the same as above and the proposed on-line/off-line threshold signature scheme consists of the following efficient algorithms:

(1) **Key generation (performed once)**
Inputs: A threshold signature scheme $\mathsf{TS} = (\mathsf{Thresh\text{-}Key\text{-}Gen}, \mathsf{Thresh\text{-}Sig}, \mathsf{Ver})$, a set of n players $\mathscr{P} = \{p_1, p_2, \ldots, p_n\}$, a threshold $t < \frac{n}{2}$, and a security parameter $l \in N$.
Public Output: A set of public keys.
Private Output: All players $p_j \in \mathscr{P}$ receive a set of private keys.

(a) On input 1^l, run the $\mathsf{Thresh\text{-}Key\text{-}Gen}$ to obtain the signing/verification key pair (SK, VK) and each $p_j \in \mathscr{P}$ receives the signing key share SK_i.
(b) Use the DKG protocol to create the *long-term* hash key $HK = Y = xP$, where $x \in \mathbb{Z}_q$ is the *long-term* trapdoor key and $p_j \in \mathscr{P}$ receives the share x_j for a degree t polynomial $R_x(z) \in \mathbb{Z}_q[z]$ such that $R_x(0) = x$.
(c) Use the DKG protocol to create k^*P, where $k^* \in \mathbb{Z}_q$ so that p_j receives the share k_j^* for a degree t polynomial $R_{k^*}(z) \in \mathbb{Z}_q[z]$ such that $R_{k^*}(0) = k^*$. Let the chameleon hash value $h = k^*P$, and run the $\mathsf{Thresh\text{-}Sig}$ to sign the message h. Let the output be $\sigma = TS_{SK}(h)$.
(d) Publish the public key (VK, Y, h, σ). All players $p_j \in \mathscr{P}$ retain (SK_j, x_j, k_j^*).

(2) **Off-line phase (performed for each message)**
In the off-line phase of Crutchfield et al.'s scheme, the output is a signature stamp σ, *i.e.*, a certain precomputed signature. However, since the chameleon hash value and the corresponding signature are always identical and published as the public key in the phase of key generation, the output in the $\mathsf{Threshold\text{-}Sig\text{-}Off\text{-}line}$ phase of our scheme is a random *one-time* trapdoor/hash key pair.[4]
Inputs: The same set of n players $\mathscr{P}$ and a threshold $t < \frac{n}{2}$.
Private output: A random *one-time* trapdoor/hash key pair.

(a) Use the DKG protocol to create k_iP, where $k_i \in \mathbb{Z}_q$ so that p_j receives the share k_{i_j} for a degree t polynomial $R_{k_i}(z) \in \mathbb{Z}_q[z]$ such that $R_{k_i}(0) = k_i$.
(b) All players $p_j \in \mathscr{P}$ retain the corresponding key pair (k_{i_j}, k_iP).

[4] Certainly, all players $p_j \in \mathscr{P}$ can precompute plenty of such key pairs in the off-line phase for the future use. In the on-line phase, the subset of $\mathscr{P}$ for signing a message can randomly retrieve such a key pair with a same index i.

(3) **On-line phase (performed for each message)**
Inputs: A subset $\mathscr{P}' \subset \mathscr{P}$ of size $t+1$ and a message $m \in \mathbb{Z}_q$.
Public output: A signature for m.

(a) For each $p_j \in \mathscr{P}'$, define $r_{i_j} = k_j^* - f(m, k_iP)(k_{i_j} + x_j) \bmod q$. Then p_j broadcasts r_{i_j} to all of the other players in $\mathscr{P}'$.
(b) Define $\gamma_j(x) = \prod_{p_l \in \mathscr{P}' \setminus \{p_j\}} \frac{l-x}{l-j}$, as in the definition of Lagrange interpolation. Now use Lagrange interpolation on the shares to compute the hash collision r_i as follows:

$$r_i = \sum_{p_j \in \mathscr{P}'} r_{i_j}\gamma_j(0) = \sum_{p_j \in \mathscr{P}'} (k_j^* - f(m, k_iP)(k_{i_j} + x_j))\gamma_j(0) \equiv k^* - f(m, k_iP)(k_i + x) \bmod q$$

(c) (r_i, k_iP) is the signature of the message m.

(4) **Verification (performed for each message)**
Inputs: A message m and the signature (r_i, k_iP).
Public Output: "Accept" or "Reject".

(a) Compute $h = f(m, k_iP)(k_iP + Y) + r_iP$ by using the *one-time* hash key k_iP and the *long-term* hash key Y.
(b) Run the Ver to check that σ is indeed a signature of the hash value h with respect to the verification key VK.

(5) **Signature share verification (performed if necessary)**
If the signature verification failed, then some players sent incorrect shares. In order to ensure robustness, we should identify and remove the corrupted players. That is, we have each player in $\mathscr{P}$ check the validity of the pair r_{i_j} for each player $p_j \in \mathscr{P}'$. Note that k_j^*P, $k_{i_j}P$, x_jP, and k_iP are known values from the DKG protocol, so for each $p_j \in \mathscr{P}'$ we can compute and conform that

$$r_{i_j}P = k_j^*P - f(m, k_iP)(k_{i_j}P + x_jP).$$

If any of the shares are deemed incorrect, then broadcast a *complaint* against p_j. If there are at least $t+1$ complaints, then clearly p_j must be corrupted since with at most t malicious players, there can be at most t false complaints. Also, if p_j is corrupted, there will always be enough honest players to generate at least $t+1$ complaints and p_j will surely be disqualified in this case. Once eliminated, p_j is removed from $\mathscr{P}'$ and is replaced with a new player, thus resulting in a new signature. As long as at most t players are corrupted, there will always be enough honest players to create a valid signature.

5.2. Security analysis

Theorem 3. *The proposed on-line/off-line threshold signature scheme $\mathscr{TS}^{\mathsf{On/Off}}$ is complete.*

Proof. Note that $r_i = k^* - f(m, k_iP)(k_i + x) \bmod q$, we have

$$h = f(m, k_iP)(k_iP + Y) + r_iP = f(m, k_iP)(k_iP + Y) + (k^* - f(m, k_iP)(k_i + x))P = k^*P$$

So, the verifier will accept the signature with probability 1. □

Theorem 4. *Suppose that an adversary corrupts at most t players, the proposed on-line/off-line threshold signature scheme $\mathscr{TS}^{\mathsf{On/Off}}$ is robust.*

Proof. The robustness of Key-Gen and Threshold-Sig-Off-line is ensured by the DKG protocol, where each player can validate his secret key share using the corresponding public verification key share.

In the phase of Threshold-Sig-On-line, if the verifier accepts the incorrect share of a corrupted player, then the corrupted player will be identified with probability 1 since the equation $r_{i_j}P = k_j^*P - f(m, k_iP)(k_{i_j}P + x_jP)$ holds for the honest players. □

Theorem 5. *The proposed on-line/off-line threshold signature scheme $\mathscr{TS}^{\mathsf{On/Off}}$ is simulatable.*

Proof. Key-Gen and Threshold-Sig-Off-line is simulatable since the DKG protocol is simulatable.

Now we prove that Threshold-Sig-On-line is simulatable. Given the system parameters SP, the message m, the corresponding signature $\sigma = (r_i, k_iP)$, t secret key shares (k_j^*, x_j) and *one-time* trapdoor/hash key pair shares (k_{i_j}, k_iP) for $1 \leqslant j \leqslant t$. The simulator can simulate the view of the adversary on an execution of Threshold-Sig-On-line as follows:

(1) Compute $r_{i_j} = k_j^* - f(m, k_iP)(k_{i_j} + x_j)$ for $1 \leqslant j \leqslant t$.
(2) Let $F(z)$ be a polynomial like function of degree t such that $F(0) = r_i$ and $F(j) = r_{i_j}$ for $1 \leqslant j \leqslant t$. Then the adversary can compute and broadcast $r_{i_j} = F(j)$ for $t+1 \leqslant j \leqslant n$. □

Table 3
Comparison of the computation cost

Key generation	Crutchfield et al.'s scheme $1C(\text{DKG})$	Our scheme $2C(\text{DKG}) + 1C(\text{Thres-Sig})$
Off-line phase	$3C(\text{DKG}) + 1m + 1C(\text{Thres} - \text{Sig})$	$1C(\text{DKG})$
On-line phase	$(2 + (2t + 1)^2)m$	$(1 + (t + 1)^2)m$
Verification	$1SM + C(\text{Ver})$	$1SM$
Signature share verification	$6M$	$1M$

Table 4
Comparison of the storage and communication cost

	Crutchfield et al.'s scheme	Our scheme
Total storage off-line phase	$2n\|q\| + 1\|\sigma\|$	$2n\|q\|$
Communication on-line phase	$2(2t + 1)\|q\| + 2\|q\| + 1\|\sigma\|$	$(t + 1)\|q\| + 2\|q\|$

Theorem 6. *If a threshold signature scheme* TS *is simulatable and the underlying signature scheme* S *is existentially unforgeable against adaptive chosen message attacks, then the threshold signature scheme* TS *is also existentially unforgeable against adaptive chosen message attacks.*

Proof. Please refer to [17]. □

Theorem 7. *Our proposed on-line/off-line threshold signature scheme* $\mathcal{TS}^{\text{On/Off}}$ *is existentially unforgeable against adaptive chosen message attacks.*

Proof. The proof can be easily deduced from Theorem 2, Theorem 4, Theorem 5, and Theorem 6. □

5.3. Efficiency analysis

We compare the efficiency of our scheme with the elliptic curve version of Crutchfield et al.'s scheme [7]. Similarly, we denote by $C(\theta)$ the computation cost of operation θ, and by $|\lambda|$ the bits of λ. Also, we denote by M a scalar multiplication in G, by SM a simultaneous scalar multiplication of the form $\lambda P + \mu Q$, and by m the modular multiplication in $\mathbb{Z}_q$. We omit other operations such as point addition and hash in both schemes.

We first analyze the computation complexity. The sum of computation cost for the key generation and off-line phase of Crutchfield et al.'s scheme is comparable to that of our scheme. While in the on-line phase of our scheme, it only requires each player $p_j \in \mathcal{P}'$ to perform 1 modular multiplication for computing a signature share, and $(t + 1)^2$ modular multiplication for the signature reconstruction. So, the computation cost of our scheme is much less than that of Crutchfield et al.'s scheme in the on-line phase. Furthermore, our scheme only requires 1 scalar multiplication for the signature share verification, which is also more efficient than Crutchfield et al.'s scheme.

We then analyze the storage and communication complexity. In the off-line phase of our scheme, each player $p_j \in \mathcal{P}$ should store $2|q|$ bits for the trapfoor/hash key share, so the total storage for n players is $2n|q|$ bits. In the on-line phase of our scheme, each player $p_j \in \mathcal{P}'$ broadcasts $|q|$ bits for the collision share to the other players. Note that the signature length is $2|q|$ bits, so the total communication cost is $(t + 1)|q| + 2|q|$ bits. Therefore, our scheme is superior to Crutchfield et al.'s scheme in storage and communication complexity.

Tables 3 and 4 present the comparison of the computation cost, the storage cost, and the communication cost for each message signing between Crutchfiled et al.'s scheme and our scheme.

6. Conclusions

On-line/off-line signatures are particularly useful in smart card applications. In this paper, we propose a new double-trapdoor chameleon hash family based on the discrete logarithm assumption and then apply the "hash–sign–switch" paradigm to propose a more efficient generic on-line/off-line signature scheme and threshold signature scheme. Compared with the existing schemes based on Shamir–Tauman's paradigm, the advantages of our signature scheme are the lower computation and storage cost for the off-line phase, and the lower communication cost for the on-line phase. Moreover, we prove that the proposed schemes achieve the desired security requirements.

Acknowledgements

We are grateful to the anonymous referees for their invaluable suggestions. This work is supported by National Natural Science Foundation of China (Nos. 60503006 and 60773202), NSFC-KOSEF Joint Research Project (No. 60611140543), and 973 Program (No. 2006CB303104).

References

[1] G. Ateniese, B. de Medeiros, Identity-based Chameleon Hash and Applications, FC 2004, LNCS, vol. 3110, Springer-Verlag, 2004, pp. 164–180.
[2] G. Ateniese, B. de Medeiros, On the Key-Exposure Problem in Chameleon Hashes, SCN 2004, LNCS, vol. 3352, Springer-Verlag, 2005, pp. 165–179.
[3] D. Boneh, X. Boyen, Short Signatures without Random Oracles, Advances in Cryptology-Eurocrypt 2004, LNCS, vol. 3027, Springer-Verlag, 2004, pp. 56–73.
[4] E. Bresson, D. Catalano, R. Gennaro, Improved On-line/off-line Threshold Signatures, PKC 2007, LNCS, vol. 4450, Springer-Verlag, 2007, pp. 217–232.
[5] D. Boneh, B. Lynn, H. Shacham, Short Signatures from the Weil Pairings, Advances in Cryptology-Asiacrypt 2001, LNCS, vol. 2248, Springer-Verlag, 2001, pp. 514–532.
[6] D. Chaum, H. van Antwerpen, Undeniable Signatures, Advances in Cryptology-Crypto 1989, LNCS, vol. 435, Springer-Verlag, 1989, pp. 212–216.
[7] C. Crutchfield, D. Molnar, D. Turner, D. Wagner, Generic On-line/Off-line Threshold Signatures, PKC 2006, LNCS, vol. 3958, Springer-Verlag, 2006, pp. 58–74.
[8] X. Chen, F. Zhang, K. Kim, Chameleon Hashing without Key Exposure, ISC 2004, LNCS, vol. 3225, Springer-Verlag, 2004, pp. 87–98.
[9] X. Chen, F. Zhang, W. Susilo, Y. Mu, Efficient Generic On-line/Off-line Signatures without Key Exposure, ACNS 2007, LNCS, vol. 4521, Springer-Verlag, 2007, pp. 18–30.
[10] Y. Desmedt, Y. Frankel, Threshold Cryptosystems, Advances in Cryptology-Crypto 1989, LNCS, vol. 435, Springer-Verlag, 1990, pp. 307–315.
[11] T. ElGamal, A public-key cryptosystem and a signature scheme based on discrete logarithms, IEEE Transactions on Information Theory 31 (4) (1985) 469–472.
[12] S. Even, O. Goldreich, S. Micali, On-line/Off-line Digital Signatures, Advances in Cryptology-Crypto 1989, LNCS, vol. 2442, Springer-Verlag, 1989, pp. 263–277.
[13] S. Even, O. Goldreich, S. Micali, On-line/off-line digital signatures, Journal of Cryptology 9 (1) (1996) 35–67.
[14] A. Fiat, A. Shamir, How to Prove Yourself: Practical Solutions to Identification and Signature Problems, Advances in Cryptology-Crypto 1986, LNCS, vol. 263, Springer-Verlag, 1986, pp. 186–194.
[15] P. Feldman, A practical scheme for non-interactive verifiable secret sharing, in: Proceeding of the 28th Annual Symposium on Foundations of Computer Science, 1987, pp. 427–437.
[16] R. Gennaro, Multi-trapdoor Commitments and their Applications to Proofs of Knowledge Secure under Concurrent Man-in-the-Middle Attacks, Advances in Cryptology-Crypto 2004, LNCS, vol. 3152, Springer-Verlag, 2004, pp. 220–236.
[17] R. Gennaro, S. Jarecki, H. Krawczyk, T. Rabin, Robust Threshold DSS Signatures, Advances in Cryptology-Eurocrypt 1996, LNCS, vol. 1070, Springer-Verlag, 1996, pp. 354–371.
[18] R. Gennaro, S. Jarecki, H. Krawczyk, T. Rabin, Secure Distributed Key Generation for Discrete-log Based Cryptosystems, Advances in Cryptology-Eurocrypt 1999, LNCS, vol. 1592, Springer-Verlag, 1999, pp. 295–310.
[19] S. Goldwasser, S. Micali, R. Rivest, A digital signature scheme secure against adaptive chosen-message attacks, SIAM Journal on Computing 17 (2) (1988) 281–308.
[20] N. Koblitz, Elliptic curve cryptosystems, Mathematics of Computation 48 (177) (1987) 203–209.
[21] H. Krawczyk, T. Rabin, Chameleon hashing and signatures, in: Proceeding of the 7th Annual Network and Distributed System Security Symposium, 2000, pp. 143–154.
[22] K. Kurosawa, K. Schmidt-Samoa, New On-line/Off-line Signature Schemes without Random Oracles, PKC 2006, LNCS, vol. 3958, Springer-Verlag, 2006, pp. 330–346.
[23] A.K. Lenstra, E.R. Verheul, Selecting cryptographic key sizes, Journal of Cryptology 14 (4) (2001) 255–293.
[24] V. Miller, Uses of Elliptic Curves in Cryptography, Advances in Cryptology-Crypto 1985, LNCS, vol. 218, Springer-Verlag, 1986, pp. 417–426.
[25] T. Pederson, A Threshold Cryptosystem without a Trusted Party, Advances in Cryptology-Eurocrypt 1991, LNCS, vol. 547, Springer-Verlag, 1991, pp. 522–526.
[26] D. Pointcheval, J. Stern, Security arguments for digital signatures and blind signatures, Journal of Cryptography 13 (3) (2000) 361–396.
[27] C.P. Schnorr, Efficient signature generation for smart cards, Journal of Cryptology 4 (3) (1991) 239–252.
[28] Z. Shao, A provably secure short signature scheme based on discrete logarithms, Information Sciences 177 (2007) 5432–5440.
[29] A. Shamir, Y. Tauman, Improved Online/Offline Signature Schemes, Advances in Cryptology-Crypto 2001, LNCS, vol. 2139, Springer-Verlag, 2001, pp. 355–367.
[30] F. Zhang, X. Chen, W. Susilo, Y. Mu, A New Signature Scheme Without Random Oracles From Bilinear Pairings, VIETCRYPT 2006, LNCS, vol. 4341, Springer-Verlag, 2006, pp. 67–80.