Efficient online/offline signcryption without key exposure

Fei Yan, Xiaofeng Chen* and Yinghui Zhang

State Key Laboratory of Integrated Service Networks (ISN), Xidian University, Xi’an 710071, China
Email: yan-wang.1987@163.com
Email: xfchen@xidian.edu.cn
Email: yhzhaang@163.com
*Corresponding author

Abstract: An online/offline signcryption scheme provides confidentiality and authentication simultaneously, and it is particularly suitable for the application of resource-constrained systems. In this paper, we present a key-exposure free online/offline signcryption scheme. In contrast, it seems that all the existing online/offline signcryption schemes based on Shamir-Tauman’s paradigm suffer from the key exposure problem. In the random oracle model, the proposed scheme is proved to be indistinguishable against adaptive chosen-ciphertext attacks (IND-CCA2) and existentially unforgeable against chosen-message attacks (EUF-CMA). Besides, an improved scheme is proposed, which requires none of the recipient’s public information in the offline phase and hence makes practical sense.

Keywords: online/offline signcryption; key exposure; chameleon hash function.


Biographical notes: Fei Yan received her Master’s degree in Engineering from Xidian University in 2012. Her research interests include cloud security and cryptography.

Xiaofeng Chen is currently a Professor at the School of Telecommunications Engineering, Xidian University. He received his PhD in Cryptography from the Xidian University in 2003. His research interests include public key cryptography, financial cryptography and cloud computing security.

Yinghui Zhang received his BS (2007) and MS (2010) from Nanchang Hangkong University and Xidian University, both in Mathematics. Currently, he is working towards the PhD degree in Cryptography, Xidian University. His research interests are in the areas of wireless network security, cloud security and cryptography.

1 Introduction

Two of the most important services offered by cryptography are those of providing private and authenticated communications. Zheng (1997) first proposed the primitive called signcryption in 1997, which is used to encrypt and sign data simultaneously in a single logical step, and thus can achieve the confidentiality and authentication more efficient than the sign-then-encrypt paradigm. Subsequently, plenty of constructions (An et al., 2002; Zheng, 2001; Boyen, 2003; Libert and Quisquater, 2003; Barreto et al., 2005; Chen and Malone-Lee, 2005) have been proposed for more efficient signcryption schemes.

In 2002, An et al. (2002) first introduced the notion of online/offline signcryption. The idea is to divide the signcryption generating procedure into two phases. The first phase is performed offline (without knowing the given message) and the second phase is performed online (after knowing the given message). However, An et al. (2002) did not give any concrete construction of online/offline signcryption. The first concrete scheme was proposed by Zhang et al. (2005). Recently, Wei et al. (2010) proposed an efficient signcryption scheme and its online/offline version, which can be proven secure without random oracles based on the assumption of $k + 1$ square roots. To the best of our knowledge, it seems that most of the existing online/offline signcryption schemes can be constructed by incorporating an online/offline signature scheme and a symmetric-key encryption algorithm.

Even et al. (1989) first introduced the notion of online/offline signature and they proposed a general method for converting any signature scheme into an online/offline signature scheme. Nonetheless, the method is rather impractical since it increases the size of the signature by a
quadratic factor. As noted in the work of Shamir and Tauman (2001), some signature schemes such as Fiat-Shamir, Schnorr and ElGamal (ElGamal, 1985; Fiat and Shamir, 1986; Schnorr, 1991) can be naturally partitioned into online and offline phase. However, these are particular schemes with special structures. In Crypto 2001, Shamir and Tauman (2001) proposed a new paradigm called ‘hash-sign-switch’ to convert any signature scheme into an online/offline signature scheme efficiently. However, the constructions based on the ‘hash-sign-switch’ paradigm suffer from the key exposure problem of chameleon hashing, which was firstly addressed by Ateniese and de Medeiros (2004) in the original chameleon signature schemes. That is, the signatures on different messages must use different chameleon hash values. Otherwise, the signer’s trapdoor information (i.e. the private key) will be recovered. The only solution to this problem is to compute and store plenty of different chameleon hash values and the corresponding signatures in the offline phase by the signer. Therefore, the computation and storage cost for the offline phase and the communication cost for the online phase are still a little more overload.

The current chameleon hash schemes without key exposure for chameleon signatures (Krawczyk and Rabin, 2000; Ateniese and de Medeiros, 2004; Chen et al., 2004; Ateniese and de Medeiros, 2005) are not suitable for designing efficient online/offline signature schemes. The reasons are twofolds: First, collision computation in these chameleon hash schemes usually requires the costly modular exponentiation operation. Second, though collision forgery will not reveal the signer’s trapdoor information, it allows the verifier to compute other collisions for the same hash value. In 2007, Chen et al. (2007) first proposed a special double-trapdoor hash family (which is named ‘Multiple-Collision Trapdoor Hash Families’ by Harn et al. (2010) and they proposed more constructions based on various assumptions) based on the discrete logarithm assumption and then incorporate it to construct a more efficient generic online/offline signature scheme without key exposure. Compared with the existing online/offline signature schemes, the advantages of Chen et al.’s signature scheme are the lower computation and storage cost for the offline phase, and the lower communication cost for the online phase. Thus, it is more suitable for the applications in the resources-constraint environment such as smart cards. After the initial work of Chen et al. (2007), there are plenty of related research on the online/offline signature schemes without key exposure (Chen et al., 2008; Guo and Mu, 2008; Guo et al., 2008; Sun et al., 2008; Gao et al., 2009; Liu and Zhou, 2009; Harn et al., 2010). It seems that most of the current online/offline signcryption schemes based on Shamir’s paradigm suffer from the key-exposure problem.

Our contributions: In this paper, with Chen et al.’s (2007) special double-trapdoor chameleon hashing as the primary ingredient, we propose an efficient online/offline signcryption scheme, which enjoys more security benefits than the existing schemes due to the key-exposure freeness. Formal security proofs are provided to support our scheme, against adaptive chosen-plaintext attacks (IND-CCA2) and adaptive chosen-message attacks (EUF-CMA), in the random oracle model, under the intractability assumption of the discrete logarithm problem. Additionally, a more practical scheme is presented, which needs none of the recipient’s public information in the offline phase.

Organisation: Our paper proceeds as follows: Some preliminaries are given in Section 2. Section 3 reviews the definition of the online/offline signcryption and the security model. Section 4 presents the proposed online/offline signcryption schemes, followed by formal security proofs and the performance analysis. Finally, we conclude this paper in Section 5.

2 Preliminaries

In this section, we will briefly describe some basic definitions and security properties with respect to our scheme.

2.1 Bilinear pairings

Let $G_1$ be a cyclic additive group generated by $P$, whose order is a prime $q$, and $G_2$ be a cyclic multiplicative group of the same order $q$. Let $a$ and $b$ be elements of $Z_q^*$. A bilinear pairing is a map $e : G_1 \times G_1 \rightarrow G_2$ with the following properties:

- **Bilinear:** $e(aR,bQ) = e(R,Q)^{ab}$ for all $R, Q \in G_1$ and $a, b \in Z_q^*$.
- **Non-degenerate:** There exists $R$ and $Q \in G_1$ such that $e(R, Q) \neq 1$.
- **Computable:** There is an efficient algorithm to compute $e(R, Q)$ for all $R, Q \in G_1$.

2.2 A new double-trapdoor chameleon hash family

Definition 1 (Double-trapdoor chameleon hash family) (Chen et al., 2007; Chen et al., 2008): A double-trapdoor chameleon hash family consists of a triple $(I_1, I_2, H)$:

1. $I_1$ 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 and can be used repeatedly during its life span.
2. $I_2$ 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 function and can be used only once.
3. $H$ is a family of randomised hash functions. Every hash function is $H$ associated with a hash key pair $HK = (HK_1, HK_2)$, and is applied to a message from a space $M$ and a random element from a finite space $R$. The output of the hash function $H_{HK}$ does not depend on $TK = (TK_1, TK_2)$.

A double-trapdoor chameleon hash family $(I_1, I_2, H)$ satisfies the properties as follows:

1. **Efficient**: Given a hash key $HK$ and a pair $(m, r) \in M \times R$, $H_{HK}$ is computable in polynomial time.

2. **Collision resistance**: There is no probabilistic polynomial-time algorithm $A$ that on input $HK$ outputs, with a probability which is not negligible, two pairs $(m_1, r_1), (m_2, r_2) \in M \times R$ 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 $(HK_1, TK_1) \leftarrow I_1(l^1)$, $(HK_2, TK_2) \leftarrow I_2(l^2)$ and over the random coin tosses of algorithm $A$).

3. **Trapdoor collisions**: There exists a probabilistic polynomial-time algorithm that given a pair $(HK_1, TK_1) \leftarrow I_1(l^1)$, a pair $(HK_2, TK_2) \leftarrow I_2(l^2)$, a pair $(m_1, r_1) \in M \times R$, and an additional message $m_2 \in M$ that satisfy $m_1 \neq m_2$, outputs a value $r_2 \in R$ such that:
   - $H_{HK}(m_1, r_1) = H_{HK}(m_2, r_2)$.
   - If $r_1$ is uniformly distributed in $R$ then the distribution of $r_2$ is computationally indistinguishable from uniform in $R$.

4. **Key-exposure freeness**: There is no probabilistic polynomial-time algorithm $A$ that on input a long-term hash key $HK_1$, two different one-time hash key $HK_2$ and $HK'_2$, two pairs $(m_1, r_1), (m_2, r_2) \in M \times R$ 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 $TK_1$.

A concrete construction of the double-trapdoor chameleon hash is proposed by Chen et al. (2007):

- **System parameters generation**: Let $t$ be a prime power and $E(F_t)$ an elliptic curve over finite field $F_t$. Let $\#E(F_t)$ be the number of points of $E(F_t)$, and $P$ be a point of $E(F_t)$ with prime order $q$, where $q | \#E(F_t)$. Denote $G$ the subgroup generated by $P$. Define a cryptographic secure hash function $f : Z_q \times G \rightarrow Z_q$.

Choose two random elements $k, x \in Z^*_q$, and compute $K = aP$, $Y = xP$. The public hash key is $HK = (K, Y)$, and the one-time/long-time trapdoor key is $TK = (k, x)$.

- **The hash family**: Given the hash key $HK$, the proposed chameleon hash function $H_{HK} : Z_q \times Z_q \rightarrow G$ is defined as follows:
  
  $$H_{HK}(m, r) = f(m, K) \cdot K + rY.$$

2.3. **The online/offline signature scheme without key exposure**

The online/offline signature scheme (Chen et al., 2007) consists of the four efficient algorithms:

- **System parameters generation**: Let $t$ be a prime power and $E(F_t)$ an elliptic curve over finite field $F_t$. Let $\#E(F_t)$ be the number of points of $E(F_t)$, and $P$ be a point of $E(F_t)$ with prime order $q$, where $q | \#E(F_t)$. Denote $G$ the subgroup generated by $P$. Define a cryptographic secure hash function $f : Z_q \times G \rightarrow Z_q$.

- **Key generation algorithm**:
  
  1. On input $l^1$, run the key generation algorithm of the original signature scheme $G$ to obtain the signing/verification key pair $(SK, VK)$.
  2. On input $l^k$, run the key generation algorithm of the trapdoor hash family to obtain the long-term/trapdoor key pair, denote by $HK = Y = xP, TK = x$.
  3. Choose at random $k' \in Z_q$, and compute the chameleon hash value $h = K^*Y$. Run the signing algorithm $S$ with the signing key $SK$ to sign the message $h$. Let $\sigma = S_{SK}(h)$ be the output.

The signing key is $(SK, x, k^*)$ and the verification key is $(VK, Y, \sigma)$.

- **The signcryption algorithm**:
  
  1. **Offline phase**
    - Choose $k_i \in Z_q$, and compute $k_ix^{-1} \mod q$ and $KP$.
    - Store the one-time trapdoor/hash key pair $(k, x^{-1}, K, P)$.
  2. **Online phase**
    - For a given signed message $m_i$, retrieve from the memory a random pair $(k, x^{-1}, k, P)$.
    - Compute $r_i = k' - f(m_i, k, P)k, x^{-1} \mod q$.
    - Send $(r_i, k_i, P)$ as the signature of the message $m_i$.

- **The verification algorithm**:
  
  1. Compute $h = f(m_i, k, P)k, P + rY$ by using the one-time hash key $k, P$ and the long-term hash key $Y$.
  2. Verify that $\sigma$ is indeed a signature of the hash value $h$ with respect to the verification key $VK$. 

2.4 Key-exposure problem

The online/offline signature schemes based on the Shamir-Tauman’s ‘hash-sign-switch’ paradigm suffer from the key-exposure problem. That is, if the signer applies the same hash value more than once to obtain two signatures on two different messages, the recipient can obtain a hash collision and use it to recover the signer’s trapdoor information.

In order to demonstrate the security flaw, i.e. the key-exposure problem, in online/offline signature, we take the Shamir-Tauman’s online/offline signature scheme on elliptic curve into consideration. Suppose that $H(m,r) = mP + rY$ is the chameleon hash function, where $Y = aP$. In the offline phase, the signer chooses a random tuple $(m', r') \in \mathbb{R} \times R$, and computes the chameleon hash value $h' = H(m', r') = m'P + r'Y$. Subsequently, the signer adopts a secure signature algorithm to obtain the signature $\sigma'$ on $h'$. In the online phase, suppose that the signer intends to sign a message $m_1$ for the recipient. Then it computes a collision $r_1$ corresponding to $m_1$ and sends $(r_1, \sigma')$ to the recipient as the signature of $m_1$. On receiving the signature $(r_1, \sigma')$ and $m_1$, the recipient can obtain the equation $h' = H(m_1, r_1) = m_1P + r_1Y$. If the signer utilises the same chameleon hash value $h'$ to sign another message $m_2$ again, then it computes the collision $r_2$ and sends $(r_2, \sigma')$ as the signature on $m_2$. After receiving $r_2$ and $m_2$, the recipient can obtain the equation $h' = H(m_2, r_2) = m_2P + r_2Y$. To be specific, the recipient gets $m_2P + r_2Y = m_1P + r_1Y$, that is $m_1 + ar_1 = m_2 + ar_2$. Obviously, the recipient can obtain the trapdoor value $a$ by computing $a = (m_1 - m_2)(r_1 - r_2)^{-1}$.

The only solution lies in using different chameleon hash values to sign different messages. However, the signer must compute and store plenty of different chameleon hash values and the corresponding signatures in the offline phase. Therefore, the computation and storage cost for the offline phase and the communication cost for the online phase are still a little more overload. And this violated the original intention of the online/offline scheme design.

As for the online/offline signcryption scheme (Wei et al., 2010), the key-exposure problem still exists. In the offline phase, the signer chooses $r, a \in Z_q$, and then generates the signature $\sigma = g^{\left(a^{2q} + ar\right)}$ and the corresponding session key. In the online phase, the signer computes a collision $r_i = ay - G(M_i)y + r$, and then the signer encrypts the signatures. Therefore, if the signer uses the same $r, a$ to sign another message $M_2$, the recipient can obtain $r_i = ay - G(M_i)y + r$ and $r_j = ay - G(M_j)y + r$. Therefore, the trapdoor information $y$ could be recovered by the equation $r_i - r_j = G(M_j)y - G(M_i)y$. It seems that most of the current online/offline signcryption schemes based on Shamir’s paradigm suffer from the key-exposure problem. Hence, it is necessary to design online/offline signcryption schemes without key exposure.

3 Security model and definitions

In this section, we give the security model and formal definition of an online/offline signcryption scheme.

3.1 The framework of online/offline signcryption

An online/offline signcryption scheme consists of the following Probabilistic Polynomial Time (PPT) algorithms:

1. KeyGen: $(SK_u, PK_u) \leftarrow \text{KeyGen}(1^k)$ takes the security parameter $1^k$ as the input, and outputs the private/public key pair of $U$ (Sender $S$ or Recipient $R$).
2. Offline-SignEnc: $\Delta' \leftarrow \text{Offline-SignEnc}(1^k, SK_u, PK_u)$ takes the security parameter $1^k$, the sender’s private key and the recipient’s public key as the input, and outputs the offline signcryption values.
3. Online-SignEnc: $\Delta \leftarrow \text{Online-SignEnc}(1^k, m, \Delta', SK_u, PK_u)$ takes the security parameter $1^k$, a given message, the sender’s private key and the recipient’s public key as the input, and outputs the signcryption values.
4. VerDec: $\text{VerDec}(1^k, \Delta, SK_u, PK_u)$ outputs a candidate message $m$ or $\perp$ from the signcryption values with the help of the keys $SK_u$ and $PK_u$. Only when the signcryption values are invalid $\perp$ be output.

For a valid signcryption scheme, it is always required for obvious consistency purpose that:

$$m = \text{VerDec(Online-SignEnc}(m, \Delta, SK_u, PK_u), SK_u, PK_u)$$

where

$$\Delta' = \text{Offline-SignEnc}(SK_u, PK_u).$$

3.2 Confidentiality

Confidentiality for online/offline signcryption schemes against adaptive chosen-ciphertext attacks (IND-CCA2) is defined via in the following IND-CCA2 game.

1. $CH$ runs $\text{KeyGen}(1^k)$ and generates the private/public key pairs $(SK_u, PK_u)$ and $(SK_R, PK_R)$, and then gives $A$ the public keys.
2. In this stage, $A$ makes two kinds of queries to the following oracles:
   - **Signcryption requests**: $A$ submits a message $m$ which he wants to query, $CH$ outputs the result of $\text{SignEnc}(m, SK_u, PK_u)$.
   - **Unsigncryption requests**: $A$ submits a signcryption value $\Delta$ which he wants to query, $CH$ outputs the results of $\text{VerDec}(\Delta, SK_u, PK_u)$.
3. A generates two messages \( m_0, m_1 \in M \) and sends them to CH. CH chooses a random bit \( b \in \{0,1\} \), and then computes a signcryption value \( \Delta \) of message \( m_b \). CH sends the value back to \( A \).

4. \( A \) can make some new queries as mentioned in Step 2 except for the Unsigncryption request for \( \Delta \).

5. At the end, \( A \) outputs his guess bit \( b' \) about the message corresponding to \( \Delta \) and wins if \( b' = b \).

The advantage of \( A \) is defined to be \( \text{Adv}^{\text{IND-CCA2}}(A) = \Pr\{b' = b\} - 1/2 \).

Definition 2 (Confidentiality): An online/offline signcryption scheme is said to be indistinguishable against adaptive chosen-ciphertext attacks if no PPT adversary \( A \) has a non-negligible advantage in the IND-CCA2 game.

### 3.3 Unforgeability

Unforgeability for online/offline signcryption schemes against adaptive chosen-message attacks (EUF-CMA) is defined via the following EUF-CMA game.

1. CH runs \( \text{KeyGen}(1^k) \) and generates the private/public key pairs \((SK_s, PK_s)\) and \((SK_k, PK_k)\), and then gives \( A \) the public keys.

2. In this stage, \( A \) makes two kinds of queries to the following oracles in an adaptive way:
   - **Signcryption requests**: \( A \) submits a message \( m \) which he wants to query, CH outputs the result of \( \text{SignEnc}(m, SK_s, PK_s) \).
   - **Unsigncryption requests**: \( A \) submits a signcryption value \( \Delta \) which he wants to query, CH outputs the results of \( \text{VerDec}(\Delta, SK_k, PK_k) \).

3. \( A \) produces a signcryption value \( \Delta \) and wins if \( \text{VerDec}(\Delta, SK_k, PK_k) \neq \bot \), where \( \Delta \) is not an output of the Signcryption request.

\( A \)'s advantage is defined as:

\[
\text{Adv}^{\text{EUF-CMA}}(A) = \Pr\{\text{VerDec}(\Delta, SK_k, PK_k) \in M\}.
\]

Definition 3 (Unforgeability): An online/offline signcryption scheme is said to be existentially unforgeable against adaptive chosen-message attacks if no PPT adversary \( A \) has a non-negligible advantage in the EUF-CMA game.

### 4 Our efficient online/offline signcryption schemes

In this section, we propose a new efficient online/offline signcryption scheme with the special double-trapdoor hash function mentioned in Section 2.2 serving as the ingredient.

#### 4.1 The proposed scheme I

- **System parameters generation**: Let \( G_1 \) and \( G_2 \) be groups of prime-order \( q \), and let \( e : G_1 \times G_1 \rightarrow G_2 \) be a collision free hash functions where \( \lambda \) is the block size of the symmetric cipher. Let \( t \) be a prime power and \( E(F) \) be an elliptic curve over finite field \( F \). Let \( \#E(F) \) be the number of points of \( E(F) \), and \( P \) be a point of \( E(F) \) with prime order \( q \) where \( q \| E(F) \). Denote \( G \) the subgroup generated by \( P \). Define a cryptographic secure hash function \( f : Z_q \times G \rightarrow Z_q \). Given a hash key \( HK = (K, Y) \), and the chameleon hash function \( H_{hk} : Z_q \times Z_q \rightarrow G \) is defined as follows:

\[
H_{hk}(m, r) = f(m, K) \cdot K + rY.
\]

- **The key generation**

  1. On input security parameter \( 1^k \), run the key generation algorithm of the trapdoor hash family to obtain the long-term trapdoor/hash key pair \((x, Y = xP)\) as the sender’s private/public key pair.

  2. On security parameter \( 1^k \), run the key generation algorithm to obtain the private/public key pair \((w, W = wP)\) of the recipient.

  3. Choose \( k' \in Z_q, v \in Z_q \), and compute the hash value \( h = k^*y, V = vP \).

The secret key is \((x, k^*, v)\) and the public key is \((Y, h, V)\).

- **The signcryption algorithm**

  1. **Offline-SignEnc**

     - Choose \( k_1 \in Z_q \), and compute \( k_1x^{-1} \mod q \) and \( K = k_1P \).

     - Compute \( u = e(vK, W) \), and then compute \( U = H_1(u) \).

     - Store the tuple \((k_1x^{-1}, K, U)\).

  2. **Online-SignEnc**

     - For a given message \( m_0 \), retrieve a random pair \((k_1x^{-1}, K, U)\).

     - Computer \( r_i = k_i^* - f(m_1, K)k_1x^{-1} \mod q \).

     - The message encryption is done with \( U \) and a symmetric-key encryption algorithm such as AES. And then compute the ciphertext \( C = Enc_e(r_i \parallel m_1) \).

     - Send \((K, C)\) to the recipient as the signcryption of \( m_0 \).
• The VerDec algorithm
1. On receiving a tuple \((K, C)\), the recipient first computes \(u^* = e(V, K)^w\), and then computes 
   \(r_i = \text{Dec}_v(C)\), here \(U^* = H_i(u^*)\).
2. The recipient checks whether \(h = f(m_i, K) \cdot K + r_i Y\) or not.

Trivially, \(u^* = e(V, K)^w = e(vP, K)^w = e(vK, W) = u\). Therefore, the signcryption scheme above satisfies the property of correctness.

In the following, we provide the formal security proofs for the proposed scheme 1. Specifically, the unforgeability, confidentiality and key-exposure freeness are achieved by Theorems 1–3, respectively.

Theorem 1 (Unforgeability): In the random oracle model, the resulting online/offline signcryption scheme is existentially unforgeable against adaptive chosen-message attacks (EUF-CMA), under the intractability assumption of the discrete logarithm problem in \(G\).

Proof: Let \((P, aP)\) be the instance of the discrete logarithm problem to be solved where \(P\) is a generator of \(G\). The aim of \(CH\) is to compute \(a\). Suppose that \(A\) is a probabilistic algorithm that forges a signcryption value with respect to the proposed online/offline signcryption scheme by an adaptively chosen-message attack in time \(T\) with success probability \(\varepsilon\).

Public key generation: \(CH\) chooses a random integer \(b, v, w \in \mathbb{Z}_q\), and let \(Y = bP, h = b \cdot aP, V = vP\) and \(W = wP\). Then, it sets the sender’s public key as \((Y, h, V)\) and the recipient’s public key as \(W\). \(CH\) provides \(A\) with these public keys. We describe how the requests are treated by \(CH\):

• \(H_1\) requests: For a query \(H_1(u)\), \(CH\) first ensures the list \(L_1\) does not contain a tuple \((u, U)\), where the list \(L_1\) is initially set to empty. If such a tuple is found, \(CH\) answers \(U\); otherwise he chooses \(U_i \in \mathbb{Z}_q\) and responds it as the answer to the query and puts the tuple \((u, U)\) into \(L_1\).

• \(f\) requests: maintain a list \(L_2\), which is initially set to empty. For a query \((m_i, K)\), \(CH\) first ensures the list \(L_2\) does not contain a tuple \((m', K', e')\). \(CH\) answers \(e\) if such a tuple is found, otherwise he chooses \(e \in \mathbb{Z}_q\) and responds it as the answer to the query and puts the tuple \((m', K', e')\) into \(L_2\).

• Signcryption requests: On receiving a message \(m'\) from \(A\), \(CH\) responds with the corresponding signcryption \((K', C')\). It is computed as follows:
  1. Choose at random \((e', r') \in \mathbb{Z}_q \times \mathbb{Z}_q\) (note that \(e'\) is not in the \(L_2\)) and \(K' = e'^{-1}(h - r'Y)\).
  2. Respond \(e'\) as the answer to the \(f\) request \((m', K')\), and add \((m', K', e')\) into the list \(L_2\).

3. Compute \(u = e(vK', W)\) (note that \(u\) is not in the \(L_1\)).
4. Respond a random \(U\) as the answer to query of the oracle \(H_1(u)\) and add \((u, U)\) into the list \(L_1\).
5. Compute \(C' = \text{Enc}_v(r'||m')\).
6. \(CH\) responds the ciphertexts \((K', C')\) to \(A\) as the answer to the query.

• Unsingcryption requests: On receiving signcryption ciphertexts \((K', C')\), \(CH\) first computes \(u = e(vK', W)\) and \(U = H_i(u)\), and then computes the signcryption \(r'||m' = \text{Dec}_v(C')\). If \(f(m', K') \cdot K' + r'Y \neq h\), \(CH\) returns \(m'\) to \(A\), otherwise reports Failure.

Eventually, \(A\) outputs a valid signcryption ciphertext \((K, C)\) for a message \(m\). \(CH\) computes \(u = e(vK, W)\), and extracts \(r\|m\) from the ciphertext \(C\) by computes \(r\|m = \text{Dec}_v(C)\). If \(m \neq m_i\) for \(j = 1, \ldots, q\) and \(h = f(m, K) \cdot K + rY\), we say that \(A\) forges a signcryption value \((K, C)\) on the special recipient with respect to the proposed online/offline signcryption scheme.

As mentioned in the work of Pointcheval and Stern (2000) and Chen et al. (2007), we can obtain two valid signatures \((m, K, r)\) and \((m, K, r')\) with respect to different hash oracles \(f\) and \(f'\). Note that \(h = f(m, K) \cdot K + rY\) and \(h = f'(m, K) \cdot K + r'Y\), we can compute \(a = (f'(m, K) - f(m, K))/r - (f'(m, K)r - f(m, K)r')\) as the discrete logarithm of \(aP\) with respect to the base \(P\).

The success probability of \(CH\) is \(\varepsilon\), and the running time of \(CH\) is equal to the running time of the Forking Lemma which is bounded by \(23T_{\text{gpl}}\varepsilon\)

Theorem 2 (Confidentiality): In the random oracle model, the resulting online/offline signcryption scheme is indistinguishable against adaptive chosen-ciphertext attacks (IND-CCA2), under the intractability assumption of the discrete logarithm problem in \(G\).

Proof: Give a random instance \((a, aP)\) of Discrete Logarithm Problem (DLP). The aim of \(CH\) is to compute \(a\).

Public key generation: \(CH\) chooses a random integer \(b, v, w \in \mathbb{Z}_q\), and lets \(Y = bP, h = b \cdot aP, V = vP\) and \(W = wP\). Then, it sets the sender’s public key as \((Y, h, V)\) and the recipient’s public key as \(W\). \(CH\) provides \(A\) with these public keys. We describe how the requests are treated by \(CH\):

• \(H_1\) requests: For a query \(H_1(u)\), \(CH\) first ensures the list \(L_1\) does not contain a tuple \((u, U)\), where the list \(L_1\) is initially set to empty. If such a tuple is found, \(CH\) answers \(U\); otherwise he chooses \(U_i \in \mathbb{Z}_q\) and responds it as the answer to the query and puts the tuple \((u, U)\) into \(L_1\).

• \(f\) requests: Maintain a list \(L_2\), which is initially set to empty. For a query \((m_i, K)\), \(CH\) first ensures the list \(L_2\) does not contain a tuple \((m', K', e')\). \(CH\) answers \(e\) if such a tuple is found, otherwise he chooses \(e \in \mathbb{Z}_q\) and responds it as the answer to the query and puts the tuple \((m', K', e')\) into \(L_2\).

• Signcryption requests: On receiving a message \(m'\) from \(A\), \(CH\) responds with the corresponding signcryption \((K', C')\). It is computed as follows:
  1. Choose at random \((e', r') \in \mathbb{Z}_q \times \mathbb{Z}_q\) (note that \(e'\) is not in the \(L_2\)) and \(K' = e'^{-1}(h - r'Y)\).
  2. Respond \(e'\) as the answer to the \(f\) request \((m', K')\), and add \((m', K', e')\) into the list \(L_2\).

3. Compute \(u = e(vK', W)\) (note that \(u\) is not in the \(L_1\)).
4. Respond a random \(U\) as the answer to query of the oracle \(H_1(u)\) and add \((u, U)\) into the list \(L_1\).
5. Compute \(C' = \text{Enc}_v(r'||m')\).
6. \(CH\) responds the ciphertexts \((K', C')\) to \(A\) as the answer to the query.

• Unsingcryption requests: On receiving signcryption ciphertexts \((K', C')\), \(CH\) first computes \(u = e(vK', W)\) and \(U = H_i(u)\), and then computes the signcryption \(r'||m' = \text{Dec}_v(C')\). If \(f(m', K') \cdot K' + r'Y \neq h\), \(CH\) returns \(m'\) to the \(A\), otherwise reports Failure.

Eventually, \(A\) outputs a valid signcryption ciphertext \((K, C)\) for a message \(m\). \(CH\) computes \(u = e(vK, W)\), and extracts \(r\|m\) from the ciphertext \(C\) by computes \(r\|m = \text{Dec}_v(C)\). If \(m \neq m_i\) for \(j = 1, \ldots, q\) and \(h = f(m, K) \cdot K + rY\), we say that \(A\) forges a signcryption value \((K, C)\) on the special recipient with respect to the proposed online/offline signcryption scheme.

As mentioned in the work of Pointcheval and Stern (2000) and Chen et al. (2007), we can obtain two valid signatures \((m, K, r)\) and \((m, K, r')\) with respect to different hash oracles \(f\) and \(f'\). Note that \(h = f(m, K) \cdot K + rY\) and \(h = f'(m, K) \cdot K + r'Y\), we can compute \(a = (f'(m, K) - f(m, K))/r - (f'(m, K)r - f(m, K)r')\) as the discrete logarithm of \(aP\) with respect to the base \(P\).

The success probability of \(CH\) is \(\varepsilon\), and the running time of \(CH\) is equal to the running time of the Forking Lemma which is bounded by \(23T_{\text{gpl}}\varepsilon\)
does not contain a tuple \((m_n, K_n, e_n)\). If such a tuple is found, \(CH\) answers \(e_n\); otherwise he chooses \(e_n \in Z_q\) and responds it as the answer to the query and puts the tuple \((m_n, K_n, e_n)\) into \(L_2\).

- **Signcryption requests:** On receiving a message \(m'\) from \(A\), \(CH\) responds with the corresponding signcryption value \((K', C')\). It is computed as follows:

  1. Choose at random \((e', r') \in Z_q \times Z_q\) (note that \(e'\) is not in the \(L_2\)) and compute \(K' = e'^{-1}(h - r'Y)\). Respond \(e'\) as the answer to query of the oracle \(f\), and add \((m', K', e')\) into the list \(L_2\).

  2. Compute \(u = e(vK, W)\) (note that \(u\) is not in the \(L_1\)), respond a random \(U\) as the answer to query of the oracle \(H_1(u)\) and add \((u, U)\) into list \(L_1\).

  3. Compute \(C = Enc_\pi(r'\|m')\).

  4. Send the ciphertexts \((K', C')\) to \(A\).

- **Unsigncryption requests:** On receiving a signcryption \((K', C')\), \(CH\) first computes \(u = e(vK, W)\), and then makes the query to the oracle \(H_1(u)\). If there exists a tuple \((u, U)\) in the list, then computes signcryption \(r'\|m' = Dec_\pi(C')\), and returns \(m'\) to the \(A\). Otherwise reports Failure.

Challenged ciphertext: \(A\) random chooses two messages \(m_0, m_1\) and sends them to the \(CH\). The only limitation is that \(m_0\), \(m_1\) have never been queried to the signcryption oracle. \(CH\) chooses randomly \(b \in \{0, 1\}\) and computes as follows:

  1. Choose at random \((e_b, r_b) \in Z_q \times Z_q\) and compute \(K_b = e_b^{-1}(h - r_bY)\).

  2. Compute \(u = e(vK, W)\), and then choose a random \(U\). Add \((u, U)\) into \(L_1\).

  3. Compute \(C_b = Enc_{\pi}(r_b \| m_b)\).

  4. Send the challenged signcryption \((K_b, C_b)\) to \(A\).

\(A\) can further access the signcryption and unsigncryption requests if the number of queries has not exceeded, with a limitation that \((K_b, C_b)\) cannot be queried to the unsigncryption request. Finally, \(A\) outputs \(b' \in \{0, 1\}\). The probability that occurs will be discussed.

If \(A\) wins the game, i.e. \(b' = b\), there must exist a tuple \((u, U)\) in the list \(L_1\), by which the ciphertext can be decrypted correctly. In this case, we will have \(h = f(m, K_b) \cdot K_b + rY\) and \(h = f(m, K_b) \cdot K_b + rY\). Then, we can compute \(a = (f(m, K_b) - f(m, K_b))^{-1}(f(m, K_b) - f(m, K_b))r_i\) as the discrete logarithm of \(aP\) with respect to the base \(P\).

**Theorem 3 (Key-exposure freeness):** The proposed scheme I enjoys the security benefit key-exposure freeness.

**Proof:** Suppose that there are two signatures on two different messages with the same chameleon hash value, and then we can obtain two equations:

\[ h = f(m_1, K_1) \cdot K_1 + r_1Y \quad \text{and} \quad h = f(m_2, K_2) \cdot K_2 + r_2Y. \]

Then

\[ f(m_1, K_1) \cdot K_1 + r_1Y = f(m_2, K_2) \cdot K_2 + r_2Y, \]

That is

\[ Y = xP\left[f(m_2, K_2) - f(m_1, K_1)\right](r_1 - r_2)^{-1} = \left[f(m_2, K_2) - f(m_1, K_1)\right]P(r_1 - r_2)^{-1}. \]

In order to extract the trapdoor information \(x\), the one-time private key \(k_1\) and \(k_2\) must be possessed at the same time. However, \(k_1\) and \(k_2\) are random numbers, and other users can only obtain the one-time public key \(K_1\) and \(K_2\). Accordingly, the trapdoor information \(x\) will not be exposed under the intractability assumption of the discrete logarithm problem in \(G\).

### 4.2 The proposed scheme II

As shown above, scheme I requires the recipient’s information to compute the session keys. However, it is impractical to compute all session keys for all potential recipients during the offline phase, which is introduced by Liu et al. (2010). In this section, we propose an improved scheme which does not require the recipient’s public information in the offline stage, and also the scheme is key-exposure free. Although the computation and the storage are a little overload, the advantage makes significant sense in real-world applications.

- **System parameters generation:** Let \(G_1\) and \(G_2\) be groups of prime-order \(q\), and let \(e : G_1 \times G_1 \rightarrow G_2\), \(H_1 : \{0, 1\}^* \rightarrow \{0, 1\}^q\) are a collision free hash functions where \(\lambda\) is the block size of the symmetric cipher. Define a cryptographic secure chameleon hash function \(H_{\text{ch}}(W, R) : G \times R \rightarrow Z_q\). Let \(t\) be a prime power and \(E(F)\) be an elliptic curve over finite field \(F_t\). Let \(|E(F)|\) be the number of points of \(E(F)\), and \(P\) be a point of \(E(F)\) with prime order \(q\) where \(q \nmid |E(F)|\). Denote \(G\) the subgroup generated by \(P\). Define a cryptographic secure hash function \(h : Z_q \times G \rightarrow Z_q\). Given a hash key \(HK = (K, Y)\), the chameleon hash function \(H_{\text{ch}} : Z_q \times Z_q \rightarrow G\) is defined as:

\[ H_{\text{ch}}(m, r) = f(m, K) \cdot K + rY. \]

- **The key generation**

  1. On input security parameter \(1_k\), run the key generation algorithm of the trapdoor hash family to obtain the long-term trapdoor/hash key pair \((x, Y = xP)\) as the sender’s private/public key pair.
2 On security parameter $1^k$, run the key generation algorithm to obtain the private/public key pair $(w, W = wP)$ of the recipient.

3 Choose $k^* \in_r \mathbb{Z}_q, v \in_r \mathbb{Z}_q$, and compute the hash value $h = k^*Y, V = vP$.

The secret key is $(x, k^*, v)$ and the public key is $(Y, h, V)$.

- **The signcryption algorithm**

  1. **Offline-SignEnc**
     - Choose $i \in_r \mathbb{Z}_q$, and compute $1 \mod ikxq$ and $iK_kP$.
     - Random choose $R \in_r \mathbb{R}$ and $W^*$, and compute $h_k(W^*, R)$.
     - Compute $U = e(vK, h_k(W^*, R))P$, and then compute $U = H_i(u)$.
     - Store the tuple $(ikx^1, K, U, W^*, R)$.

  2. **Online-SignEnc**
     - For a given message $m_i$, retrieve a random pair $(ikx^1, K, U, W^*, R)$.
     - Compute $r_i = k^* - f(m_i, K)x^1 \mod q$.
     - Compute $R'$ from the equation $h_k(W^*, R) = h_k(e(W, xh), R')$.
     - The message encryption is done with $U$ and a symmetric-key encryption algorithm such as AES. And then compute the signcryption $C = Enc_U(r_i || m_i)$.
     - Send $(K, C, R')$ to the recipient as the signcryption of the message $m_i$.

- **The VerDec algorithm**

  1. On receiving a tuple $(K, C, R')$, the recipient computes $u' = e(V, K)^{h_k(e(W, xh), R')}$ and $r_j \parallel m_j = Dec_U(C)$, here $U' = H_i(u')$.

  2. The recipient checks whether $h = f(m_j, K) \cdot K + r_jY$ or not.

Trivially, we have

$$u' = e(V, K)^{h_k(e(W, xh), R')} = e(V, K)^{h_k(e(W, vP), R')} = e(V, K)^{h_k(e(W', vP), R')}.$$  

Therefore, the signcryption scheme II satisfies the property of correctness.

Taking a method similar to that used for scheme I, the formal security proofs of scheme II can also be obtained.

### 4.3 Efficiency and property analysis

We compare the efficiency and security property of our schemes with that of Wei et al.’s (2010) scheme in Tables 1 and 2, respectively. We denote by $P$ the pair computation, by $Exp$ the exponent computation, and by $|\lambda|$ the bit length of $\lambda$. 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 $\mathbb{Z}_q$.

Obviously, comparing with Wei et al.’s (2010) scheme, both our schemes enjoy more security benefits due to the key exposure freeness. Additionally, the proposed scheme I is most efficient in terms of the computation cost in different phases and the offline storage overheads. The proposed scheme II has the advantage of needing none of the recipient’s public information in the offline stage at the cost of a little sacrifice of performance.

### Tables

#### Table 1

<table>
<thead>
<tr>
<th>Schemes</th>
<th>Wei et al. (2010)</th>
<th>Proposed scheme I</th>
<th>Proposed scheme II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offline cost</td>
<td>$1P + 3Exp$</td>
<td>$1P + 1M + 1m$</td>
<td>$1P + 1M + 1m$</td>
</tr>
<tr>
<td>Online cost</td>
<td>$1m$</td>
<td>$1m$</td>
<td>$1P + 1m$</td>
</tr>
<tr>
<td>Unsigncryption cost</td>
<td>$2P + 3Exp$</td>
<td>$1P + 1SM$</td>
<td>$2P + 1SM$</td>
</tr>
<tr>
<td>Offline storage</td>
<td>$2</td>
<td>q</td>
<td>+</td>
</tr>
<tr>
<td>Ciphertext size</td>
<td>$l_{c_i} + l_m + l_r$</td>
<td>$l_{c_i} + l_m + l_r$</td>
<td>$l_{c_i} + l_m + 2l_r$</td>
</tr>
</tbody>
</table>

#### Table 2

<table>
<thead>
<tr>
<th>Schemes</th>
<th>Wei et al. (2010)</th>
<th>Proposed Scheme I</th>
<th>Proposed Scheme II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidentiality</td>
<td>IND-CCA2</td>
<td>IND-CCA2</td>
<td>IND-CCA2</td>
</tr>
<tr>
<td>Unforgeability</td>
<td>EUF-CMA</td>
<td>EUF-CMA</td>
<td>EUF-CMA</td>
</tr>
<tr>
<td>Key-exposure freeness</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
5 Conclusions

We have constructed a new online/offline signcryption scheme overcoming the key exposure problem of chameleon hashing. Compared with the existing online/offline signcryption scheme, our scheme is significantly more efficient during the online stage. Moreover, our scheme is proven to be IND-CCA2 and EUF-CMA secure. We also propose a more practical scheme which requires none of the recipient’s public information in the offline stage and hence makes practical sense.

Acknowledgements

We are grateful to the anonymous referees for their invaluable suggestions. This work is supported by the National Natural Science Foundation of China (Nos. K5051010001, K50510010003 and JY10000901034).

References


