Short Generic Transformation to Strongly Unforgeable Signature in the Standard Model*

Joseph K. Liu[1], Man Ho Au[2], Willy Susilo[2], and Jianying Zhou[1]

[1] Cryptography and Security Department
Institute for Infocomm Research, Singapore
{ksliu,jyzhou}@i2r.a-star.edu.sg
[2] Centre for Computer and Information Security (CCISR)
School of Computer Science and Software Engineering
University of Wollongong, Australia
{aau,wsusilo}@uow.edu.au

Abstract. Standard signature schemes are usually devised to merely achieve existential unforgeability, i.e., to prevent forgeries on new messages not previously signed. Unfortunately, existential unforgeability is not suitable for several applications, since a new signature on a previously signed message may be produced. Therefore, there is a need to construct signature schemes with strong unforgeability, that is, it is hard to produce a new signature on any message, even if it has been signed before by legitimate signer. Recently, there have been several generic transformations proposed to convert weak unforgeability into strong unforgeability. For instance, various generic transforms of signatures that are existential unforgeable under adaptive chosen message attack (`uf-cma`) to strongly unforgeable under adaptive chosen message attack (`suf-cma`) have been proposed. Moreover, methods of converting signatures that are existentially unforgeable under generic chosen message attack (`uf-gma`) to `uf-cma` secure digital signatures have also been studied. Combination of these methods yields generic transform of digital signatures offering `uf-gma` security to `suf-cma` security. In this paper, we present a short universal transform that directly converts any `uf-gma` secure signatures into `suf-cma` secure. Our transform is *the shortest generic transformation*, in terms of signature size expansion, which results in `suf-cma` secure signature in the standard model. While our generic transformation can convert any `uf-gma` secure signature to `suf-cma` secure signature directly, the efficiency of ours is comparable to those which only transform signatures from `uf-gma` secure to `uf-cma` secure in the standard model.

1 Introduction

Digital signatures are amongst the most fundamental primitive of modern cryptography. The definition on the security of digital signatures was first formally

* The first and forth author of this work are funded by the A*STAR project SEDS-0721330047.

D. Gritzalis, B. Preneel, and M. Theoharidou (Eds.): ESORICS 2010, LNCS 6345, pp. 168–181, 2010.

introduced by Goldwasser, Micali and Rivest [14]. Security of digital signatures can be classified according to the goals, as well as resources available to the adversary. As defined in [14], the goals of the adversary can be classified into four categories, listed below in descending order of difficulty.

- *Total Break*: The adversary is able to output the secret key of the signer.
- *Universal Forgery*: The adversary is able produce forgery on *any* message.
- *Selective Forgery*: The adversary is able to forge a signature on a message chosen before he learns the public key of the signer.
- *Existential Forgery*: The adversary is able to forge a signature on a new message[1] of its choice.

Security notions of digital signatures are also classified according to the resources available to the adversary. Two kinds of attacks are defined in [14]. In a *Key-Only Attack*, the adversary is only given the public key of the signer. *Message Attack* allows the enemy to examine some signatures corresponding to either known or chosen-messages prior to his attempt to break the scheme. *Message Attack* are further sub-divided into four categories, listed below in ascending order of adversary's capabilities.

- *Known Message*: The adversary is given access to signatures for a set of messages. These messages are known to the adversary but are *not* chosen by him.
- *Generic Chosen Message*: Also known as *Weak Chosen Message Attack* in [6] - the adversary is given access to signatures for a set of messages chosen by him. However, the adversary must choose the set of messages *before* knowing the public key of the signer.
- *Directed Chosen Message*: Similar to *Generic Chosen Message*, the adversary is provided with access to signatures for a set of messages chosen by him except he can now select the set *after* knowing the public key of the signer.
- *Adaptive Chosen Message*: This is the most general "natural" attack that any adversary can mount: the adversary is allowed to use the signer as an "oracle" in which he can request signatures from the signer of message on its choice[2].

Nowadays, standard signature schemes are usually designed to achieve existential unforgeability under adaptive chosen message attack (`uf-cma`). That is, forgery of signatures on new messages not previously signed are impossible even if the adversary is given signatures on any other messages of its choice. However, most signature schemes are randomized and allow many possible signatures for a single message. In this case, the notion `uf-cma` security of signature does not rule out

[1] By "new", we mean the corresponding signature of this message has not been provided to the adversary.

[2] The choice may depend on the signer's public key as well as previously obtained signatures.

the possibility of producing a new signature on a previously signed message. In other words, the adversary, given a message/signature pair (m, σ), maybe able to forge a new valid signature $\sigma' \neq \sigma$ on the same message m. In some applications, a stronger security notion, called *strong unforgeability* [1], is desirable.

Strong unforgeability ensures the adversary cannot even produce a new signature for a previously signed message. Consequently, any signature that passes the verification algorithm must be coming from the legitimate signer.

Strongly unforgeable signatures are useful in many different kinds of applications, such as building chosen-ciphertext secure encryption schemes [12,11], group signatures [2,7], signcryption [1] and authenticated key exchange [16]. Unfortunately, many signature schemes in the literature are *not* strongly unforgeable.

Sometimes existential unforgeability is also referred to as weak unforgeability (as in [21]) to emphasize its difference with strong unforgeability.

1.1 Generic Security-Amplifying Methods for Ordinary Digital Signatures

Several existing works [8,22,21,15,5,18] studied the problem of converting a signature scheme satisfying a weaker security notion into a stronger one with the help of some seemingly simpler cryptographic primitive, such as one-time signature or chameleon hash function. For instance, various generic transforms of signatures that are existential unforgeable under adaptive chosen message attack (`uf-cma`) to strongly unforgeable under adaptive chosen message attack (`suf-cma`) have been proposed [8,22,21,15,5]. Moreover, methods of converting signatures that are existentially unforgeable under generic chosen message attack (`uf-gma`) to `uf-cma` secure digital signatures have been studied in [13,17,20,6,23,18]. Combination of these methods yields generic transform of digital signatures offering `uf-gma` security to `suf-cma` security.

1.2 Our Result

We propose an efficient generic transformation for any signature scheme secure against existential forgery under generic chosen message attack (`uf-gma`) to a strongly unforgeable one under adaptive chosen message attack (`suf-cma`) in the standard model. Our transform is *universal* in the sense we treat the underlying signature as a blackbox and do not exploit any of its internal structure. Moreover, our transform only requires *one* chameleon hash function with a special property called *Given-Target One-Wayness* (GTOW). We observe that several existing chameleon hash functions exhibit this property. Therefore, we are not required to construct a specially designed chameleon hash function. Employing one of the most efficient chameleon hash functions, our transformation only increases the resulting signature by *one* group element. Hence, we refer to our transform as a *short universal transform*. Our transformation outperforms previous transformations in the following ways:

1. It is the shortest generic transformation, in terms of signature size expansion, which results in `suf-cma` secure signature in the standard model. Previous transformation either requires the random oracle model [22], or increases the signature size by at least 2 group elements [15], or only works for signature scheme with specific structure (e.g., partitioned signatures [8]).
2. While our generic transformation can convert any `uf-gma` secure signature to `suf-cma` secure signature directly, the efficiency of ours is comparable to those which only transform signatures from `uf-gma` secure to `uf-cma` secure in the standard model [13,17,20].

The features of existing transformations are summarized in Fig. 1. We compare the existing transformability (in the sense of `uf-gma, uf-cma` and `suf-cma`), signature size (if the transformation relies on another primitive such as chameleon hash function or one-time signature, we consider the most efficient one) and the security model.

It is easily observable that our transform is very efficient and only produces one group element in terms of signature size expansion.

1.3 Related Works

(`uf-cma` TO `suf-cma`): Recently, some techniques have been proposed to convert `uf-cma` secure signature schemes to `suf-cma` secure signatures. Boneh *et al.* [8] proposed the first conversion of such kind. However, their conversion is *not* universal. It can be only applied to some specific type of signatures, called *partitioned* signatures. Although many schemes fall in this class, some do not, such as DSA signature [19]. Teranishi *et al.* [22] proposed two universal conversions. Different from [8], they can be used to convert any kind of signatures. The first one requires the random oracle model and the second one works in the standard model. Both schemes rely on the hardness of the discrete-logarithm problem. The signature size increases by 1 element for the scheme secure in the random oracle model and by 2 elements for the scheme secure in the standard model. Steinfeld *et al.* [21] proposed another transformation[3] in the standard model which requires two chameleon hash functions. Similar to [22], the signature size also increases by two elements. All of the above transforms are based on chameleon hash functions or similar mechanisms. Huang *et al.* [15] proposed a generic transformation that makes use of strong one-time signature. The signature size increases by more than 2 to 3 elements, due to the public key of the one-time signature which is included in the final signature. Bellare *et al.* [5] employed a similar approach, by using a two-tier signature to produce the transformation. The resulting signature size also increases by at least 2 elements.

(`uf-gma` TO `uf-cma`): A signature which is `uf-gma` secure can be converted to `uf-cma` secure in the random oracle model, by using a hash function of the messages for signatures without changing any other algorithms. The random

[3] Their transformation is also applicable to `uf-gma` secure signature, as noted in the remark of [21].

	uf-gma	uf-cma	suf-cma	Increase in Signature Size (group element)	Standard Model	Remarks
Boneh *et al.* [8]		→		+1	✓	applicable to partitioned signatures only
Teranishi *et al.* [22] protocol 1		→		+1	×	RO only
Teranishi *et al.* [22] protocol 2		→		+2	✓	
Steinfeld *et al.* [21]	→	→		+2	✓	
Huang *et al.* [15]		→		+2	✓	
[13, 17, 20]	→			+1	✓	
Li *et al.* [18]	→			+2	✓	
Our Transformation	→	→		+1	✓	

Fig. 1. Comparison of different transformations

oracle allows the challenger to back patch the messages into different values that are fixed at the beginning [6,23].

Another approach is to use a chameleon hash function. A chameleon hash function is a kind of hash function with a trapdoor. By knowing the trapdoor information, one can find a pair of collision easily. Without this trapdoor, it is just similar to a normal hash function. The signer can first sign any value with the weak signature scheme. Then it can sign the real message from the signature on any value, by using the property of chameleon hash function with the knowledge of the trapdoor information. Both the public key and the signature size will be increased by at least one element. This technique has been used in [13,17,20].

Recently Li *et al.* [18] proposed another method to convert a `uf-gma` secure signature to a `uf-cma` secure one, by signing the message twice using the same weakly secure signature scheme. The signature size is increased by two elements.

1.4 Organization

The rest of the paper is organized as follows. We review some mathematical and security definition in Section 2. We introduce the new property *Given-Target One-Wayness* of chameleon hash function in Section 3 and give two examples to show that some existing classic chameleon hash functions possess this property. In Section 4 we present our generic transformation. Finally, we conclude the paper in Section 5.

2 Definition

2.1 Mathematical Assumption

Definition 1 (Discrete Log (DL) Assumption). *Let g be a generator in group G. Given an element $y \in G$ chosen at random, find an integer x such that $y = g^x$. An adversary $\mathcal{A}$ has at least an ϵ advantage if*

$$\Pr[\mathcal{A}(y, g, G) = x \mid y = g^x] \geq \epsilon$$

We say that the (ϵ, τ)-DL assumption holds if no algorithm running in time at most τ can solve that DL problem with advantage at least ϵ.

Definition 2 (One-More Discrete Log (OMDL) Assumption). *[4,3] Let g be a generator in group G. The adversary is given access to a discrete log solver oracle access $\mathcal{O}_{DL}$ in which when the adversary inputs an elements $y' \in G$ the oracle returns an element x' such that $y' = g^{x'}$, and a challenge oracle, which, returns a challenge point $y \in G$ each time it is invoked. Let $y_1, \ldots, y_n \in G$ be the challenges returned by the challenge oracle. The adversary is asked to output x_i such that $y_i = g^{x_i}$, for $i = 1, \ldots, n$ with the restriction that it can query $\mathcal{O}_{DL}$ for at most $n - 1$ times. An adversary $\mathcal{A}$ has at least an ϵ advantage if*

$$\Pr[\mathcal{A}^{\mathcal{O}_{DL}}(y_1, \ldots, y_n, g, G) = (x_1, \ldots, x_n) \mid y_1 = g^{x_1}, \ldots, y_n = g^{x_n}\ ] \geq \epsilon$$

We say that the (n, ϵ, τ)-OMDL assumption holds if no algorithm running in time at most τ can solve that OMDL problem with advantage at least ϵ.

2.2 Security Definition

Definition of Signature: We first review the definition of a digital signature scheme from [14] below.

Definition 3. *A signature scheme SIG is defined by three algorithms:*

- KeyGen is a probabilistic polynomial-time (PPT) algorithm that takes a security parameter k as input. It returns a public key pk and a secret key sk.
- Sign is a PPT algorithm taking as input (sk, m) where m is a message from a message space $\mathcal{S}$. It outputs a signature σ.
- Verify is a deterministic polynomial-time algorithm taking (pk, m, σ) as input. It returns either valid or invalid.

For correctness, we require that $\mathsf{valid} \leftarrow \mathsf{Verify}(pk, m, \sigma)$ where $\sigma \leftarrow \mathsf{Sign}(sk, m)$ and $(pk, sk) \leftarrow \mathsf{KeyGen}(1^k)$.

Security of Signature: We review three levels of security for a signature, namely, *existential unforgeability against generic chosen message attack* (`uf-gma`) [14], *existential unforgeability against adaptive chosen message attack* (`uf-cma`) [14] and *the strong unforgeability against adaptive chosen message attack* [1]. We start with the most common one, `uf-cma`.

- `uf-cma`: Existential unforgeability against adaptive chosen message attack can be defined using the following game called **Game-UFCMA**:
  Setup: A public/secret key pair $(pk, sk) \leftarrow \mathsf{KeyGen}(1^k)$ is generated and adversary $\mathcal{A}$ is given the public key pk.

Query: $\mathcal{A}$ runs for time t and issues q signing queries to a signing oracle in an adaptive manner. For each i, $1 \leq i \leq q$, $\mathcal{A}$ chooses a message m_i based on the message-signature pairs that $\mathcal{A}$ has already seen, and obtains in return a signature σ_i on m_i from the signing oracle. That is, $\sigma_i \leftarrow \mathsf{Sign}(sk, m_i)$.

Forge: $\mathcal{A}$ outputs a forgery (m^*, σ^*) and halts. $\mathcal{A}$ wins if

1. σ^* is a valid signature on message m^* under the public key pk. That is, $\mathsf{valid} \leftarrow \mathsf{Verify}(pk, m^*, \sigma^*)$; and
2. m^* has never been queried. That is, $m^* \notin \{m_1, \ldots, m_q\}$.

Definition 4 (uf-cma). *A signature scheme SIG=(KeyGen, Sign, Verify) is (t, q, ϵ)-existentially unforgeable against adaptive chosen message attacks, if any adversary with running time t wins the **Game-UFCMA** with probability at most ϵ after issuing at most q signing queries.*

– uf-gma: In a uf-gma game, the adversary has to submit all signature queries before seeing the public key. We refer to this game as **Game-UFGMA** which is defined as follows:

Query: The adversary $\mathcal{A}$ submits a list of q messages $m_1, \ldots, m_q$.

Setup: $\mathcal{A}$ is given a public key pk together with a list of message-signature pairs (m_i, σ_i) for $1 \leq i \leq q$.

Forge: Same as in **Game-UFCMA**.

Definition 5 (uf-gma). *A signature scheme SIG=(KeyGen, Sign,Verify) is (t, q, ϵ)-existentially unforgeable against generic chosen message attacks, if any adversary with running time t wins the **Game-UFGMA** with probability at most ϵ after issuing at most q signing queries.*

– suf-cma: One of the restrictions for adversary $\mathcal{A}$ in **Game-UFCMA** (resp. **Game-UFGMA**) is that the forging message m^* must be new. We can relax this restriction to obtain the notion of ***strong*** *existential unforgeable against adaptive chosen message attacks*, such that $\mathcal{A}$ forges a new valid signature on a message that could have been signed previously. We refer to this new game as **Game-SUFCMA** which is defined as follows:

Setup: Same as in **Game-UFCMA**.

Query: Same as in **Game-UFCMA**.

Forge: $\mathcal{A}$ outputs a forgery (m^*, σ^*) and halts. $\mathcal{A}$ wins if

1. σ^* is a valid signature on message m^* under the public key pk. That is, $\mathsf{valid} \leftarrow \mathsf{Verify}(pk, m^*, \sigma^*)$; and
2. $(m^*, \sigma^*) \notin \{(m_1, \sigma_1) \ldots, (m_q, \sigma_q)\}$.

Definition 6 (suf-cma). *A signature scheme SIG=(KeyGen, Sign,Verify) is (t, q, ϵ)-strongly existentially unforgeable against adaptive chosen message attacks, if any adversary with running time t wins the **Game-SUFCMA** with probability at most ϵ after issuing at most q signing queries.*

Chameleon Hash Function: In our generic transformation, we also require another primitive called the *chameleon hash function* [17,20]. We first describe the basic property of a chameleon hash function here, and introduce a new property in the next section. A chameleon hash function is a special type of hash function, whose collision resistance depends on the user's state of knowledge. Every chameleon hash function is associated with a pair of public key and private key, referred to as the hash key HK and the trapdoor key TK, respectively:

Definition 7 (Chameleon Hash Family). *A chameleon hash family consists of the following algorithms:*

- $\mathsf{HKey\text{-}Gen}$ is a PPT key generation algorithm that on input a security parameter k, outputs a pair (HK, TK).
- Hash_{HK} is a deterministic polynomial-time algorithm that on input a hash key HK and a pair $(M, R) \in \mathcal{M} \times \mathcal{R}$ where $\mathcal{M}$ is the message space and $\mathcal{R}$ is the space of the random element, output a hashed value C.
- $\mathsf{Collision\text{-}Finding}$ is a deterministic polynomial-time algorithm that on input a trapdoor and hash key pair (TK, HK), a pair $(M, R) \in \mathcal{M} \times \mathcal{R}$, and an additional message $M' \in \mathcal{M}$, outputs a value $R' \in \mathcal{R}$ such that $\mathsf{Hash}_{HK}(M, R) = \mathsf{Hash}_{HK}(M', R')$.

3 Given-Target One-Way Chameleon Hash

We introduce a new property called *given-target one-wayness* to chameleon hash functions. Informally speaking, it means that given a hash value, an attacker cannot find a pre-image without the trapdoor even if it is given another pre-image.

We examine some existing chameleon hash functions and show that at least two of the existing constructions satisfy this newly proposed property.

3.1 Definition

We define the property with the following game.

Definition 8 (Given-Target One-Wayness). *A chameleon hash family is possessing (t, q, ϵ) given-target one-wayness if no PPT adversary $\mathcal{A}$ with running time t, making at most q query can win in the following game with probability ϵ:*

Setup: *A hash key and trapdoor key pair $(HK, TK) \leftarrow \mathsf{HKey\text{-}Gen}(1^k)$ is generated and adversary $\mathcal{A}$ is given the hash key HK.*

Given-Target One-Way Query: *For the i-th query, $\mathcal{A}$ is given a hash value h_i. $\mathcal{A}$ submits a value M_i and is given R_i such that $\mathsf{Hash}_{HK}(M_i, R_i) = h_i$. Queries can be adaptive and interleaving.*

Output: *$\mathcal{A}$ outputs (M^*, R^*) and wins if $\mathsf{Hash}_{HK}(M^*, R^*) = h_i$ for some i and that $(M^*, R^*) \neq (M_i, R_i)$.*

We say a chameleon hash function is *given-target one-way* (GTOW) if it possesses given-target one-wayness.

3.2 Scheme under DL Assumption

We review the double trapdoor chameleon hash from [10] and show that it is GTOW under the discrete logarithm assumption.

The Scheme Scheme_{DL}: HKey-Gen. Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order p. Randomly generate $x, y \in_R \mathbb{Z}_p$ and compute $u = g^x$, $v = g^y$. The message space $\mathcal{M}$ is $\mathbb{Z}_p$. The trapdoor key TK is (x, y) and the hash key HK is $(\mathbb{G}, p, g, u, v, \mathcal{M})$.

Hash_{HK}. For a message $M \in \mathcal{M}$, choose a random $\tilde{R} \in_R \mathbb{Z}_p$, $R \in_R \mathbb{Z}_p$ and compute

$$C = u^M v^{\tilde{R}} g^R$$

Collision-Finding. For a given pair $(M, \tilde{R}, R)$ and a message $M' \in \mathcal{M}$, in order to find $(\tilde{R}', R')$ such that $\mathsf{Hash}_{HK}(M, \tilde{R}, R) = \mathsf{Hash}_{HK}(M', \tilde{R}', R')$, first randomly generates $\tilde{R}' \in_R \mathbb{Z}_p$, then use the trapdoor key TK to compute

$$R' = (M - M')x + (\tilde{R} - \tilde{R}')y + R \bmod p$$

Lemma 1 (Scheme_{DL} is GTOW under DL assumption). *Assume there is an adversary $\mathcal{A}$ who can break the (τ, q, ϵ) given-target one-wayness of Scheme_{DL}, we can construct a simulator $\mathcal{S}$ to solve the (ϵ', τ')-DL problem, where*

$$\epsilon' \geq \frac{1}{2}\epsilon \qquad \tau = \tau'$$

Proof. Suppose there exists an adversary $\mathcal{A}$ in the sense of Definition 8. We construct a simulator $\mathcal{S}$, having black-box access to $\mathcal{A}$, that solves the DL problem.

Setup. $\mathcal{S}$ is given $\mathbb{G} = \langle w \rangle$ such that $|w| = p$ for some prime p and a value Y, and its goal is to output $y \in \mathbb{Z}_p$ such that $Y = w^y$. $\mathcal{S}$ randomly generates $x \in_R \mathbb{Z}_p$ and flips a fair coin $b \in_R \{0, 1\}$. If $b = 0$, computes $u = w^x$ and sets $v = Y$ and $g = w$. Otherwise, computes $u = w^x$ and sets $v = w$ and $g = Y$. $\mathcal{S}$ gives $(\mathbb{G}, p, g, u, v, \mathcal{M} := \mathbb{Z}_p)$ to $\mathcal{A}$ as HK of the trapdoor hash.

Query. For the i-th query, $\mathcal{S}$ randomly generates $m_i', \tilde{r}_i', r_i' \in_R \mathbb{Z}_p$, computes $h_i = u^{m_i'} v^{\tilde{r}_i'} g^{r_i'}$ and returns h_i to $\mathcal{A}$. Upon receiving the corresponding m_i from $\mathcal{A}$, $\mathcal{S}$ computes $r_i = (m_i' - m_i)x + r_i'$ and sets $\tilde{r}_i = \tilde{r}_i'$ if $b = 0$. Otherwise, $\mathcal{S}$ computes $\tilde{r}_i = (m_i' - m_i)x + \tilde{r}_i'$ and sets $r_i = r_i'$. Then, $\mathcal{S}$ returns $(\tilde{r}_i, r_i)$ to $\mathcal{A}$.

Output. Finally, $\mathcal{A}$ outputs $m^*, \tilde{r}^*, r^*$ such that $u^{m^*} v^{\tilde{r}^*} g^{r^*} = u^{m_i} v^{\tilde{r}_i} g^{r_i}$ for some i. If $b = 0$, $\mathcal{S}$ aborts if $\tilde{r}^* = \tilde{r}_i$. Otherwise $\mathcal{S}$ computes

$$y = \frac{(m_i - m^*)x + r_i - r^*}{\tilde{r}^* - \tilde{r}_i}$$

If $b = 1$, $\mathcal{S}$ aborts if $r^* = r_i$. Otherwise, $\mathcal{S}$ computes

$$y = \frac{(m_i - m^*)x + \tilde{r}_i - \tilde{r}^*}{r^* - r_i}$$

$\mathcal{S}$ returns y as the solution of the discrete logarithm problem.

Successful Simulation. Since $(m^*, \tilde{r}^*, r^*) \neq (m_i, \tilde{r}_i, r_i)$, at least one of the statements $\tilde{r}^* \neq \tilde{r}_i$ or $r^* \neq r_i$ is true. For each case, with probability $1/2$, $\mathcal{S}$ does not abort. The running time of $\mathcal{S}$ is just the same as $\mathcal{A}$. □

3.3 Scheme under One-More-DL Assumption

Next, we review the classic DL-based chameleon hash function [9,17] and show that it is GTOW under the one-more discrete logarithm assumption.

The Scheme Scheme_{OMDL}: $\mathsf{HKey\text{-}Gen}$. Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order p. Randomly generate $x \in_R \mathbb{Z}_p$ and compute $u = g^x$. The message space $\mathcal{M}$ is $\mathbb{Z}_p$. The trapdoor key TK is (x) and the hash key HK is $(\mathbb{G}, p, g, u, \mathcal{M})$.

Hash_{HK}. For a message $M \in \mathcal{M}$, choose a random $R \in_R \mathbb{Z}_p$ and compute

$$C = u^M g^R$$

$\mathsf{Collision\text{-}Finding}$. For a given pair (M, R) and a message $M' \in \mathcal{M}$, in order to find (R') such that $\mathsf{Hash}_{HK}(M, R) = \mathsf{Hash}_{HK}(M', R')$, use the trapdoor key TK to compute

$$R' = (M - M')x + R \bmod p$$

Lemma 2 (Scheme_{OMDL} is GTOW under OMDL assumption). *Assume there is an adversary $\mathcal{A}$ who can break the (τ, q, ϵ) given-target one-wayness of Scheme_{OMDL}, we can construct a simulator $\mathcal{S}$ to solve the $(q-1, \epsilon', \tau')$-OMDL problem, where*

$$\epsilon' = \epsilon \qquad \tau = \tau'$$

Proof. Suppose there exists an adversary $\mathcal{A}$ in the sense of Definition 8. We construct a simulator $\mathcal{S}$, having black-box access to $\mathcal{A}$, that solves the one-more discrete logarithm problem.

Setup. $\mathcal{S}$ is given $\mathbb{G} = \langle g \rangle$ such that $|g| = p$ for some prime p access to two oracles, namely, $\mathcal{O}_Q$ and $\mathcal{O}_{DL}$. When invoked for the i-th time, $\mathcal{O}_Q$ returns an element $Y_i \in \mathbb{G}$ while $\mathcal{O}_{DL}(X_i)$ returns $x_i \in \mathbb{Z}_p$ such that $X_i = g^{x_i}$. The goal of $\mathcal{S}$ is to solves all the discrete logarithm of Y_i to base g while using less number of query to $\mathcal{O}_{DL}$. $\mathcal{S}$ invokes $\mathcal{O}_Q$, obtains Y_0 and sets $u = Y_0$. $\mathcal{S}$ gives $(\mathbb{G}, p, g, u, \mathcal{M} := \mathbb{Z}_p)$ to $\mathcal{A}$ as HK of the trapdoor hash.

Query. For the i-th query, $\mathcal{S}$ invokes $\mathcal{O}_Q$ to receive Y_i. $\mathcal{S}$ then sets $h_i = Y_i$ and returns h_i to $\mathcal{A}$. Upon receiving the corresponding m_i from $\mathcal{A}$, $\mathcal{S}$ invokes $\mathcal{O}_{DL}(\frac{h_i}{u^{m_i}})$ and obtains r_i such that $h_i = u^{m_i} g^{r_i}$. $\mathcal{S}$ returns r_i to $\mathcal{A}$.

Output. Finally, $\mathcal{A}$ outputs m^*, r^* such that $u^{m^*} g^{r^*} = u^{m_i} g^{r_i}$ for some i. $\mathcal{S}$ computes $y_0 = \frac{(r_i - r^*)}{m^* - m_i}$. Next, $\mathcal{S}$ computes $y_i = m_i y_0 + r_i$. $\mathcal{S}$ returns y_0 as the discrete logarithm of Y_0 and y_i's as the discrete logarithms of Y_i.

Successful Simulation. Clearly the successful probability and running time of $\mathcal{S}$ are the same as $\mathcal{A}$. □

4 Generic Transformation to SUF-CMA Secure Signature

4.1 Our Transformation

We build a generic and universal transformation which converts any `uf-gma` secure signature scheme $\text{SIG}' = (\mathsf{KeyGen}', \mathsf{Sign}', \mathsf{Verify}')$ (with message space $\mathcal{S}'$) to a `suf-cma` secure signature scheme SIG=(KeyGen, Sign, Verify). This transformation requires a give-target one-way chameleon hash function $\mathsf{Hash}_{HK} : \mathcal{M} \times \mathcal{R} \rightarrow \mathcal{S}'$ (we use the notation described in Section 2.2) and a regular collision-resistant hash function $H : \{0,1\}^* \rightarrow \mathcal{M}$. The transformation is as follows:

- KeyGen: Generate a public / secret key pair $(pk', sk') \leftarrow \mathsf{KeyGen}'(1^k)$. Generate a hash key and trapdoor key $(HK, TK) \leftarrow \mathsf{HKey\text{-}Gen}(1^k)$. Set the public key as $pk = (pk', HK, \mathsf{Hash}_{HK}, H)$ and secret key as $sk = (sk', TK)$.
- Sign: On input a secret key sk and a message m, the following steps are carried out and a signature σ is generated:
  1. Randomly generate $m' \in_R \mathcal{M}$ and $r' \in_R \mathcal{R}$.
  2. Compute $h' \leftarrow \mathsf{Hash}_{HK}(m'; r')$.
  3. Compute $s' \leftarrow \mathsf{Sign}'(sk', h')$.
  4. Compute $h \leftarrow H(m||s')$.
  5. Using the trapdoor key TK to find r such that $\mathsf{Hash}_{HK}(h; r) = \mathsf{Hash}_{HK}(m'; r')$.
  6. Output the signature $\sigma \leftarrow (s', r)$.
- Verify: On input public key pk, message m and signature σ, output

$$\mathsf{Verify}'\Big(pk', \mathsf{Hash}_{HK}\Big(H(m||s'); r\Big), s'\Big)$$

4.2 Security Analysis

Theorem 1 (Strong Unforgeability). *Assume there is an adversary $\mathcal{A}$ who can break the (τ, q, ϵ) `suf-cma` security of the signature scheme, we can construct a simulator $\mathcal{S}$ to either break the $(\tau, q, \epsilon/2)$ `uf-gma` security of the underlying signature scheme, or the $(\tau, q, \epsilon/2)$ given-target one-wayness of the chameleon hash function Hash_{HK}.*

Proof. Suppose there exists an adversary $\mathcal{A}$, making q signing queries, wins Game-SUFCMA with non-negligible probability with SIG=(KeyGen, Sign, Verify), we show that either $\text{SIG}' = (\mathsf{KeyGen}', \mathsf{Sign}', \mathsf{Verify}')$ is not `uf-gma` secure or Hash_{HK} is not given-target one-way.

The proof is by reduction. Firstly, we classify the adversary $\mathcal{A}$ into two types. Suppose the q message-signature pairs $\mathcal{A}$ obtained in the Query phase are $\big(m_i,$

$(s'_i, r_i)\big)$ for $i = 1$ to q and the final forged message-signature pair is $\big(m^*, (s'^*, r^*)\big)$. We say $\mathcal{A}$ is of type I if $\mathsf{Hash}_{HK}\big(H(m^*||s'^*); r^*\big) = \mathsf{Hash}_{HK}\big(H(m_i||s'_i); r_i\big)$ for some $i = 1$ to q. Otherwise, $\mathcal{A}$ is of type II.

Next, we construct a simulator $\mathcal{S}$ which simulates GAME-SUFCMA for type I and II adversary differently, in such a way that the adversary cannot distinguish which kind of simulation it is in.

1. Type I Adversary (Breaking Given-Target One-wayness Property of Hash_{HK})
   (Setup.) $\mathcal{S}$ is given a chameleon hash function $\mathsf{Hash}_{HK} : \mathcal{M} \times \mathcal{R} \to \mathcal{S}'$ without the trapdoor and its goal is to break given-target one-wayness as defined in Definition 8. It invokes KeyGen' and obtains (pk', sk'). Let H be a collision-resistant hash function. $\mathcal{S}$ gives $\mathcal{A}$ $(pk', HK, \mathsf{Hash}_{HK}, H)$ as the public key of the signature scheme SIG.
   (Query.) For a signature query m, $\mathcal{S}$ issues a Given-Target One-Way Query and obtains h_i. It sets $h' = h_i$ and computes $s' = \mathsf{SIG}'(sk', h')$ using the corresponding secret key sk'. It computes $h = H(m||s')$ and sets $m_i = h$. It sends m_i to the Given-Target One-Way Oracle and receives r_i such that $h_i = \mathsf{Hash}_{HK}(m_i, r_i)$. It sets $r = r_i$ and returns (s', r) as the corresponding signature.
   (Forgery.) Finally, $\mathcal{A}$ outputs a message m^* and a corresponding signature (s'^*, r^*) such that $\mathsf{Verify}'(pk', \mathsf{Hash}_{HK}(H(m^*||s'^*; r))), s'^*)$ returns valid.
   (Reduction.) Since $\mathcal{A}$ is of type I, there exists a query message m_i with resulting signature (s'_i, r_i) such that

$$\mathsf{Hash}_{HK}\big(H(m^*||s'^*); r^*\big) = \mathsf{Hash}_{HK}\big(H(m_i||s'_i); r_i\big).$$

   $\mathcal{S}$ returns $(H(m^*||s'^*), r^*)$ as the output that breaks the given-target one-wayness of Hash_{HK}.
2. Type II Adversary (Breaking `uf-gma` Security of SIG′)
   (Setup.) $\mathcal{S}$ invokes $\mathsf{HKey\text{-}Gen}(1^k)$ and obtains (HK, TK) for the chameleon hash function Hash_{HK}. $\mathcal{S}$ randomly generates m'_i, r'_i and computes $h'_i = \mathsf{Hash}_{HK}(m'_i; r'_i)$ for $i = 1$ to q where q is the total number of signature query. $\mathcal{S}$ submits $h'_1, \cdots, h'_q$ to the challenger as the chosen messages and obtains a signature scheme SIG′ with public key pk' and signatures of s'_i for $i = 1$ to q as the signature of h'_i under public key pk'. $\mathcal{S}$ gives $\mathcal{A}$ $(pk', HK, \mathsf{Hash}_{HK}, H)$ as the public key of the signature scheme SIG.
   (Query.) For the i-th query, $\mathcal{S}$ is given a message m_i. It computes $h_i = H(m_i||s'_i)$ and find r_i such that $\mathsf{Hash}_{HK}(h_i; r_i) = h'_i$ using the trapdoor TK. Return (s'_i, r_i) to $\mathcal{A}$ as the corresponding signature.
   (Forgery.) Finally, $\mathcal{A}$ outputs a message m^* and a corresponding signature (s'^*, r^*) such that $\mathsf{Verify}'(pk', \mathsf{Hash}_{HK}(H(m^*||s'^*; r))), s'^*)$ returns valid.
   (Reduction.) $\mathcal{S}$ computes $M^* = \mathsf{Hash}_{HK}(H(m^*||s'^*; r)))$ and returns (M^*, s'^*) as the forged message-signature pair to the underlying signature scheme SIG′.

For the probability analysis, it is easy to see that for each type it can be perfectly simulated, so that each type occurs with the same probability as in the real attack. Hence either the attacker of type I or type II succeeds with probability $\epsilon/2$, as claimed. The running time should be the same. □

5 Conclusion

We presented an efficient universal transform that directly converts any `uf-gma` secure signatures into `suf-cma` secure. Our transform is the shortest generic transformation, in terms of signature size expansion. While our generic transformation can convert any `uf-gma` secure signature to `suf-cma` secure signature directly, the efficiency of ours is comparable to those which only transform signatures from `uf-gma` secure to `uf-cma` secure in the standard model. We make use of a new property in chameleon hash functions called *given-target one-wayness*. We demonstrated that this property exists in at least two of the existing constructions of chameleon hash functions in the literature, and henceforth, we do not need to create a specially designed chameleon hash function to be incorporated in our transform.

References

1. An, J., Dodis, Y., Rabin, T.: On the security of joint signatures and encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 83–107. Springer, Heidelberg (2002)
2. Ateniese, G., Camenisch, J., Joye, M., Tsudik, G.: A practical and provably secure coalition-resistant group signature scheme. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 255–270. Springer, Heidelberg (2000)
3. Bellare, M., Namprempre, C., Pointcheval, D., Semanko, M.: The one-more-rsa-inversion problems and the security of chaum's blind signature scheme. J. Cryptology 16(3), 185–215 (2003)
4. Bellare, M., Palacio, A.: Gq and schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 162–177. Springer, Heidelberg (2002)
5. Bellare, M., Shoup, S.: Two-tier signatures, strongly unforgeable signatures, and Fiat-Shamir without random oracles. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 201–216. Springer, Heidelberg (2007)
6. Boneh, D., Boyen, X.: Short Signatures Without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
7. Boneh, D., Boyen, X., Shacham, H.: Short group signatures using strong Diffie-Hellman. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)
8. Boneh, D., Shen, E., Waters, B.: Strongly unforgeable signatures based on computational Diffie-Hellman. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 229–240. Springer, Heidelberg (2006)
9. Boyar, J., Kurtz, S.A., Krentel, M.W.: A discrete logarithm implementation of perfect zero-knowledge blobs. J. Cryptology 2(2), 63–76 (1990)

10. Bresson, E., Catalano, D., Gennaro, R.: Improved on-line/off-line threshold signatures. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 217–232. Springer, Heidelberg (2007)
11. Canetti, R., Halevi, S., Katz, J.: Chosen-ciphertext security from identity-based encryption. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)
12. Dolev, D., Dwork, C., Naor, M.: Non-malleable cryptography. SIAM J. Computing 30(2), 391–437 (2000)
13. Even, S., Goldreich, O., Micali, S.: On-line/off-line digital signatures. J. Cryptology 9(1), 35–67 (1996)
14. Goldwasser, S., Micali, S., Rivest, R.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. Computing 17(2), 281–308 (1998)
15. Huang, Q., Wong, D.S., Zhao, Y.: Generic transformation to strongly unforgeable signatures. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 1–17. Springer, Heidelberg (2007)
16. Katz, J., Yung, M.: Scalable protocols for authenticated group key exchange. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 110–125. Springer, Heidelberg (2003)
17. Krawczyk, H., Rabin, T.: Chameleon signatures. In: NDSS, The Internet Society (2000)
18. Li, J., Kim, K., Zhang, F., Wong, D.S.: Generic security-amplifying methods of ordinary digital signatures. In: Bellovin, S.M., Gennaro, R., Keromytis, A.D., Yung, M. (eds.) ACNS 2008. LNCS, vol. 5037, pp. 224–241. Springer, Heidelberg (2008)
19. National Institute of Standards and Technology (NIST). Digital Signature Standard (DSS). Federal Information Processing Standards Publication 186-2 (January 2000)
20. Shamir, A., Tauman, Y.: Improved online/offline signature schemes. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 355–367. Springer, Heidelberg (2001)
21. Steinfeld, R., Pieprzyk, J., Wang, H.: How to strengthen any weakly unforgeable signature into a strong unforgeable signature. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 357–371. Springer, Heidelberg (2006)
22. Teranishi, I., Oyama, T., Ogata, W.: General conversion for obtaining strongly existentially unforgeable signatures. In: Barua, R., Lange, T. (eds.) INDOCRYPT 2006. LNCS, vol. 4329, pp. 191–205. Springer, Heidelberg (2006)
23. Zhang, F., Safavi-Naini, R., Susilo, W.: An efficient signature scheme from bilinear pairings and its applications. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 277–290. Springer, Heidelberg (2004)