Improving Security of q-SDH Based Digital Signatures

Fuchun Guo[☆], Yi Mu, Willy Susilo

Centre for Computer and Information Security Research
School of Computer Science and Software Engineering
University of Wollongong, Wollongong NSW2522, Australia
{fg278,ymu,wsusilo}@uow.edu.au

Abstract

In Eurocrypt 2009, Hohenberger and Waters pointed out that a complexity assumption, which restricts the adversary to a single correct response, seems inherently more reliable than their flexible counterparts. The q-SDH assumption is less reliable than standard assumptions because its solution allows exponential answers. On the other hand, the q-SDH assumption exhibits the nice feature of tight reducion in security proof. In this paper, we propose a variant of the q-SDH assumption, so that its correct answers are polynomial and no longer exponentially many. The new assumption is much more reliable and weaker than the original q-SDH assumption. We propose a new digital signature scheme that can *tightly* reduce the security to the proposed assumption in the standard model. We show that our signature scheme shares most properties with the q-SDH based signature schemes. We also propose a new approach to construct fully secure signatures from weakly secure signature against known-message attacks. Although our security transformation is conditional and not completely generic, it offers another efficient approach to construct fully secure signatures.

Keywords:
Digital Signatures, q-SDH Assumption, Security Proof

1. Introduction

Digital signatures are a central primitive in modern cryptography. The security proof of a signature scheme is usually based on a complexity assumption. The q-Strong Diffie-Hellman (q-SDH) assumption [4] has become

[☆]Corresponding Author

a common assumption used for a catalogue of signatures. Digital signatures [4, 5, 12] based on the q-SDH assumption have many good properties. For example, their public key are short with strong unforgeability [4]; their security proofs can be in the standard model [14] without random oracles [2, 7]; their security reductions are tight [13]. Compared to digital signatures based on the CDH assumption [20, 16], these q-SDH based signatures give the nice feature of tight notion in security proof. We refer the reader to [12, 3] for the importance of tight reduction.

The q-SDH assumption was first defined by Boneh and Boyen in [4]. Roughly speaking, the q-SDH assumption in a bilinear group $\mathbb{G}$ of prime order p states that it is intractable to compute $(c, g^{1/(a+c)})$ for a freely chosen integer $c \in \mathbb{Z}_p$[1], where the input is $g, g^a, g^{a^2}, \cdots, g^{a^q} \in \mathbb{G}$. Although its time complexity is less than standard assumptions [8, 5], we can choose a larger group size to increase its time complexity. In this paper, we are not interested in its time complexity but a comment made by Hohenberger and Waters [16], who pointed out that the q-SDH assumption is less reliable than standard assumptions due to the number of correct answers. As the integer c can be freely chosen by the adversary from the space $\mathbb{Z}_p$, correct answers for a challenge input of the q-SDH assumption are exponentially many, while the correct answers for those standard assumptions have one only. For example, the only correct answer for the CDH assumption is $g^{ab} \in \mathbb{G}$ for a challenge input $g, g^a, g^b \in \mathbb{G}$. Standard assumptions restricting an adversary with one correct answer seem inherently "harder" than the q-SDH assumption, which allows an adversary with a flexibility to win [16].

Restricting c in choice for the adversary is the only way to fill this gap. If the integer c is given and fixed in the challenge input, then there is only one correct answer. Especially, when c is restricted with $c = 0$ and attached in the challenge input, this specific q-SDH assumption is equivalent to the q-DHI assumption [5]. The q-DHI assumption, or q-Diffie-Hellman Inversion assumption, states that it is intractable to output $g^{1/x}$ with the same input as the q-SDH assumption. However, it is hard to tightly reduce the security of digital signatures to the q-DHI assumption. Consider the simulation approaches in [4, 5, 12] using the q-SDH assumption. Informally speaking, the integer c is decided by the inverted exponent[2] and is related to signed messages. The simulator can solve the q-SDH assumption using the forged

[1] By convention in this context we define $1/0$ to be 0 so that in the unlikely event that $a + c = 0$ we define $g^{1/(a+c)} = 1_{\mathbb{G}}$ [5].

[2] We name b_2 in g^{b_1/b_2} for $b_1, b_2 \in \mathbb{Z}_p$ as an inverted exponent.

signature but the message in this forged signature is decided by the adversary. As a result, the integer c of the q-SDH assumption is decided by the adversary. This is the reason why we cannot tightly reduce the security to the q-DHI assumption.

In this paper, we define a variant of q-SDH assumption whose correct answers are limited in a smaller range (i.e., polynomial), instead of exponentially many. In this variant, the integer c of the answer $(c, g^{1/(a+c)})$ for the q-SDH assumption is restricted in the range $[1, R]$. We define it as q-SDH_1^R assumption. The original q-SDH assumption can then be denoted by q-SDH_1^p. Suppose $R = 2^{15}$ and $p \approx 2^{160}$, we have that the q-SDH_1^R assumption with 2^{15} correct answers is more reliable than the q-SDH assumption with exponentially many answers.

Let q_B be the upper bound of the number of signatures generated from each private key and suppose the number of queried signatures in the security proof is no more than q_s ($q_s \leq q_B$). We propose a new signature scheme whose security can be reduced to the q-$\mathrm{SDH}_1^{\sqrt{q_B}}$ assumption, such that q is the smaller value of $\{q_s + 1, \sqrt{q_B} + 1\}$. Our assumption with smaller parameter q and fewer correct answers is definitely *harder* than the $(q_s + 1)$-SDH assumption used in [4, 5, 12]. Our construction still shares most of nice properties of the digital signatures [4, 5, 12] that are based on the q-SDH assumption, such as random oracle freeness, tight reduction and short public key. Our approach to realize this construction is based on the idea of length-restricted bit strings and stateful signing, which are originally from [15] and [16] respectively. Our scheme can restrict the adversary in forging a valid signature, so that we can program the security reduction to the q-$\mathrm{SDH}_1^{\sqrt{q_B}}$ assumption.

To realize our signature construction, we introduce a new security transformation. It is shown in [19, 4] that digital signatures that are secure against *week* chosen-message attacks can be converted into *adaptive* chosen-message attacks (i.e., fully secure). The security transformation incorporates the chameleon hash function [18, 19]. The authors in [19] also proposed a security transformation from known-message attacks to adaptive chosen-message attacks using a particular chameleon hash function based on factoring assumptions. In this paper, with chameleon hash based on discrete log assumptions, we show that *some* digital signatures that are secure against *known-message* attacks can be converted into adaptive chosen-message attacks. In comparison with the transformation in [19] based on factoring assumptions, the new transformation based on discrete log assumptions offers a *shorter signature*. We provide some conditions for this transformation

to be applicable. Although our security transformation is conditional, it demonstrates a new technique to enable the construction of fully secure signature schemes.

The rest of the paper is organized as follows. We give definitions in Section 2, and define the q-SDH$_1^R$ complexity assumption in Section 3. In Section 4, we give our security transformation. In Section 5, we present our signature. In Section 6, we conclude our work.

2. Definitions

2.1. Digital Signatures

A digital signature scheme is composed of the following three algorithms.

KeyGen: On input a security parameter 1^λ, the algorithm returns a key pair (pk, sk). The public key is pk and the private key is sk. The private key size depends on the length of security parameter.

Sign: On input a message m from a message space $\mathcal{M}$ and the private key sk, the algorithm returns a signature $\sigma_{sk}[m]$.

Verify: On input a signed message $(m, \sigma_{sk}[m])$ and its public key pk, the algorithm returns *accept* or *reject*.

The definition above is a standard notion of digital signatures, which are also considered as stateless signatures [17], in contrast of stateful signatures [16]. A stateful signature scheme is different from stateless signature scheme in the Sign algorithm. A stateful scheme requires that the signer defines a piece of stateful information S, which is initialized before signing operations. Given a message m to be signed, the signer firstly updates the stateful information and computes the signature $\sigma_{sk}[m]$ using both the private key sk and the stateful information S. Let the algorithm be Sign^S for the stateful signature scheme. A stateful signature consists of the following three algorithms.

KeyGen: It is the same as the stateless signature. Let S be a piece of stateful information. S is initialized before signature computations.

Sign^S: On input a message m from a message space $\mathcal{M}$, the stateful information S and the private key sk, the algorithm updates the stateful information S and returns a signature $\sigma_{sk}[m]$.

Verify: It is the same as the stateless signature.

The stateful information S plays an important role in the Sign^S algorithm. Observe that the stateful construction (e.g. [16]) is no longer secure when two different signatures use the same stateful information S in generations. Hence, besides the private key, the stateful information S should be protected against external modification. This is also required in our scheme.

2.2. Security Model

The standard security notion for digital signatures is *existentially unforgeable against adaptive chosen-message attacks*, which is considered as *fully secure*. This notion is firstly defined by Goldwasser, Micali and Rivest in [14]. They also proposed a weaker security notion known as *existentially unforgeable against known-message attacks* or KMA secure. The security notions are defined using a game between a challenger and an adversary. The difference between fully secure and KMA secure is the choice of messages in the query phase. In the standard model, messages to be queried are adaptively chosen by the adversary, while in the KMA model, they are chosen by the challenger. We here briefly revisit their definitions.

The model of existential unforgeability with respect to adaptive chosen-message attacks is defined as follows.

Setup: The challenger runs the algorithm **KeyGen** to obtain a key pair (pk, sk). The public key pk is forwarded to the adversary.

Query: The adversary queries the signature on the message m_i, which is adaptively chosen by itself. The challenger responds the query by running Sign algorithm and returning the resulting $\sigma_{sk}[m]$. Here, we use q_s to denote the number of queried signatures.

Output: The adversary outputs a forged signature $(m^*, \sigma_{sk}[m^*])$, and wins the game if no query was made on m^* and the signature $\sigma_{sk}[m^*]$ is correct.

Definition 1. *A digital signature scheme is (t, q_s, ϵ)-fully secure if there exists no adversary $\mathcal{A}$, who can run in t time at most, make q_s signature queries adaptively at most and forge a correct signature with probability ϵ at least.*

The model of existential unforgeability with respect to known-message attacks is defined as follows.

Setup: The same as the Setup phase of the standard model.

Query: The adversary queries signatures. Let q_s be the number of signature queries. The challenger responds by returning signed messages $(m_1, \sigma_{sk}[m_1])$,

$(m_2, \sigma_{sk}[m_2]), \cdots, (m_{q_s}, \sigma_{sk}[m_{q_s}])$ to the adversary, where all messages are chosen by the challenger.

Output: The same as the Output phase of the standard model.

Definition 2. *A digital signature scheme is (t, q_s, ϵ)-KMA secure if there exists no adversary $\mathcal{A}$, who can run in t time at most, receive q_s signed messages at most and forge a correct signature with probability ϵ at least.*

The above KMA attack model is also called as the *random message attack* due to the introduction in [19], when all messages are random. This notion is defined to complete a security transformation to adaptive chosen-message attacks using a particular chameleon hash based on factoring assumptions. Our security transformation also requires the known messages to be random, or is based on signature schemes secure against random message attacks. The comparison will be presented in Section 4.

The original security notions in [14] are concerned with stateless signatures only. They can be naturally extended for stateful signatures, by replacing the algorithm Sign with Sign^S. Since the security models are similar, we omit them here.

2.3. Bilinear Pairing

We briefly review the definition of bilinear pairing with the notion in [6, 4]. The bilinear pairing is composed of groups $\mathbb{G}, \mathbb{G}_T$ and a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Both $\mathbb{G}$ and $\mathbb{G}_T$ have order p and g is a generator of $\mathbb{G}$. The bilinear map e should satisfy the following two properties.

- For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
- $e(g, g)$ is a generator of $\mathbb{G}_T$ if g generate the group $\mathbb{G}$.

This bilinear pairing is referred to as symmetric pairing. A more general case is known as asymmetric pairing. Generally, the asymmetric pairing is composed of groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ and a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, such that $\mathbb{G}_1 \neq \mathbb{G}_2$. The main difference of symmetric pairing and asymmetric pairing is the representation of elements in $\mathbb{G}_1$ and $\mathbb{G}_2$. When they are specifically chosen [6], the elements in $\mathbb{G}_1$ are smaller in size than those of $\mathbb{G}_2$. We use this asymmetric pairing to construct shortest possible signatures [5], when a signature contains elements in the group $\mathbb{G}$. In this paper, we utilize only the compact symmetric pairing to describe our construction. It can be definitely replaced with an asymmetric pairing for shorter signature size.

2.4. Chameleon Hash

The notion of chameleon hash is firstly introduced by Krawczyk and Rabin in [18]. Basically, the input of a chameleon hash function is a message m and randomness r, denoted by $H_{ch}(m, r)$. The property of a chameleon hash function falls into two types depending on whether its trapdoor key τ is known or not.

- τ **is unknown**. In this case, given the input $m, r, H_{ch}(m, r)$, it is collusion-resistant for an adversary to find a different pair $(m', r') \neq (m, r)$ such that $H_{ch}(m, r) = H_{ch}(m', r')$.
- τ **is known**. In this case, given the input $m, r, H_{ch}(m, r)$, a different message m' and the trapdoor information τ, it is efficient to compute r' such that $H_{ch}(m, r) = H_{ch}(m', r')$.

Secure chameleon hash functions have been successfully constructed under different complexity assumptions, such as the discrete logarithm problem and integer factorization [19]. We review the construction of a chameleon hash using the pairing group $\mathbb{G}$ under the discrete logarithm assumption. This construction is also related to our security transformation. It is defined as follows.

Randomly choose τ from $\mathbb{Z}_p$ and compute $h = g^\tau$. The chameleon hash function is composed of public parameters g, h and the trapdoor key τ. Both message m and randomness r are chosen from $\mathbb{Z}_p$. The function $H_{ch}(m, r)$ is defined as $H_{ch}(m, r) = h^m g^r$. One may easily find that this chameleon hash function satisfies the two properties above. A detailed security proof in the standard model can be found in [19].

A chameleon hash function can also be constructed based on factoring assumptions. Shamir and Tauman also provided two constructions. Their first construction has a strong property in collusion finding that given the trapdoor key, a new message m' and an image H_{ch} (to replace with the pair (m, r)), it is efficient to compute r' such that $H_{ch} = H_{ch}(m', r')$. This property enables the conversion for signature schemes from random message attacks to adaptive chosen-message attacks. In this work, we do not use this (strong) chameleon hash function since it gives a long signature size, but instead, we utilize the above (weak) chameleon hash function based on the discrete log assumptions. More importantly, we achieve the *same* security transformation.

3. Complexity

In this section, we review q-SDH assumption and q-DHI assumption, and define the q-SDH$_1^R$ assumption. All assumptions are described in the pairing group $\mathbb{G}$.

Definition 3 (q-SDH). *Given $g, g^a, g^{a^2}, \cdots, g^{a^q} \in \mathbb{G}$ as the challenge input, the q-SDH problem is to compute a pair $(c, g^{\frac{1}{a+c}}) \in \mathbb{Z}_p \times \mathbb{G}$ for a freely chosen integer c in $\mathbb{Z}_p$.*

We say that the q-SDH problem is a (t, ϵ)-hard problem if the following probability

$$\epsilon = \Pr\left[(c, g^{\frac{1}{a+c}}) \leftarrow \mathcal{A}(g, g^a, g^{a^2}, \cdots, g^{a^q})\right]$$

is negligible over random choices of $a \in \mathbb{Z}_p$ and $g \in \mathbb{G}$ for all adversaries $\mathcal{A}$, who runs in t polynomial time at most.

Definition 4 (q-DHI). *Given $g, g^a, g^{a^2}, \cdots, g^{a^q} \in \mathbb{G}$ as the challenge input, the q-DHI problem is to compute the element $g^{\frac{1}{a}}$ in $\mathbb{G}$.*

We say that the q-DHI problem is a (t, ϵ)-hard problem if the following probability

$$\epsilon = \Pr\left[g^{\frac{1}{a}} \leftarrow \mathcal{A}(g, g^a, g^{a^2}, \cdots, g^{a^q})\right]$$

is negligible over random choices of $a \in \mathbb{Z}_p$ and $g \in \mathbb{G}$ for all adversaries $\mathcal{A}$, who runs in t polynomial time at most.

Definition 5 (q-SDH$_1^R$). *Given $g, g^a, g^{a^2}, \cdots, g^{a^q} \in \mathbb{G}$ and $R \in \mathbb{Z}_p$ as the challenge input, the q-SDH$_1^R$ problem is to compute a pair $(c, g^{\frac{1}{a+c}}) \in \mathbb{Z}_p \times \mathbb{G}$ for a freely chosen integer c satisfying $1 \leq c \leq R$.*

We say that the q-SDH$_1^R$ problem is a (t, ϵ)-hard assumption if there exists no adversary who runs in t polynomial time at most and solves the problem with non-negligible probability ϵ. Here, ϵ is defined as

$$\epsilon = \Pr\left[(c, g^{\frac{1}{a+c}}) \leftarrow \mathcal{A}(g, g^a, g^{a^2}, \cdots, g^{a^q}, R)\right]$$

over random choices of $a \in \mathbb{Z}_p$ and $g \in \mathbb{G}$.

We list the comparison of the three assumptions in the following table.

Assumptions	Challenge Input	Challenger Output	Condition
q-SDH	$g, g^a, g^{a^2}, \cdots, g^{a^q}$	$(c, g^{1/(a+c)})$	$c \in [1, p]$
q-SDH$_1^R$	$g, g^a, g^{a^2}, \cdots, g^{a^q}, R$	$(c, g^{1/(a+c)})$	$c \in [1, R]$
q-DHI	$g, g^a, g^{a^2}, \cdots, g^{a^q}$	$(c, g^{1/(a+c)})$	$c = 0$

For all the three assumptions, their answers can be modified into the same format except the choice of integer c. Let q be same in all the three assumptions and R be 2^{15}. Although the q-SDH$_1^R$ problem is "easier" than the q-DHI problem, it is much more reliable than the q-SDH problem which has exponential answers.

Remark 1. There exists a more generic definition on the q-SDH$_1^R$ assumption. We call it q-SDH$_{R_1}^{R_2}$ assumption, where the integer c of its solution is restricted in the range $[R_1, R_2]$. However, we do not use this definition since the existence of self-reduction when $R_2 - R_1 = R - 1$. Consider the instance $(g, g^a, \cdots, g^{a^q})$ as the challenger input of the q-SDH$_1^R$ problem. To reduce an answer to the q-SDH$_{R_1}^{R_2}$ problem, we can re-compute its input as $(g, g^{a+R_1}, \cdots, g^{(a+R_1)^q})$. Then, the challenger output of the modified q-SDH$_1^R$ problem is the very output for the q-SDH$_{R_1}^{R_2}$ problem.

Remark 2. Digital signatures whose security is based on the q-SDH problem or the q-SDH$_1^R$ problem can be also reduced to the q-DHI problem. The idea is similar to the self-reduction approach in the Remark 1. However, reducing a solution of the q-SDH problem to the q-DHI problem will have an exponential security loss. While the reduction from the q-SDH$_1^R$ problem to the q-DHI problem loses only $(\log R)$-bit security. Significant security loss will slow down the implementation. More details are discussed in [12, 3]. We discuss tight reduction in this paper only.

4. Security Transformation

To realize our security transformation, we firstly define a new notion, named as *compilable digital signatures*. Then, we describe our security transformation based on this specific notion using a chameleon hash function.

Definition 6. *Let $\mathcal{S} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ be a digital signature scheme and $\sigma_{sk}[m]$ be a signature on m signed with the private key sk. Let $g \in \mathbb{G}$ be a public parameter of chameleon hash. We call that $\mathcal{S}$ is a compilable signature scheme in $\mathbb{G}$ if*

- *There exists a compiler that can efficiently compile the signature $\sigma_{sk}[m]$ into $\sigma^{\star}_{sk}[g^m]$, and compile the signature $\sigma^{\star}_{sk}[g^m]$ into $\sigma_{sk}[m]$ if m is given.*

- *There exists an algorithm $\mathsf{Verify}^{\star}$ that can verify whether $\sigma^{\star}_{sk}[y]$ is a valid signature on y signed with the private key sk.*

Let $\mathcal{S}^{kma} = (\mathsf{KeyGen}^{\mathsf{kma}}, \mathsf{Sign}^{\mathsf{kma}}, \mathsf{Verify}^{\mathsf{kma}})$ be the KMA secure compilable signature scheme. Suppose $\mathsf{Verify}^{\star}$ is the verification algorithm for the signature scheme $\mathcal{S}^{kma}$. We firstly describe how to construct the fully secure signature scheme $\mathcal{S} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ from $\mathcal{S}^{kma}$.

KeyGen: Use the algorithm KeyGen^{kma} to generate a key pair (pk, sk). Let $(g, h) \in \mathbb{G}$ be the public parameter of a secure chameleon hash function and τ be its trapdoor key, such that $h = g^{\tau}$. The public key PK for the fully secure scheme is (pk, g, h) and the private key SK is (sk, τ).

Sign: Given a message m to be signed, run the algorithm Sign^{kma} to return a message-signature pair $(y, \sigma_{sk}[y])$, compile the signature $\sigma_{sk}[y]$ into $\sigma^{\star}_{sk}[g^y]$ and compute $r = y - \tau m \in \mathbb{Z}_p$. The signature on m is

$$\Sigma_{sk}[m] = (r, \sigma^{\star}_{sk}[g^y]).$$

Verify: Given the signed message $(m, r, \sigma^{\star}_{sk}[g^y])$, compute $h^m g^r$ and verify that $\sigma^{\star}_{sk}[g^y]$ is a signature on $h^m g^r$ with the algorithm $\mathsf{Verify}^{\star}$.

The following theorem shows that our security transformation is correct.

Theorem 1. $\mathcal{S} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *is fully secure if*

- *The chameleon hash function in $\mathbb{G}$ is collusion-resistant.*

- *Signed messages returned from $\mathcal{S}^{kma}$ are universally random in $\mathbb{Z}_p$ (or, $\mathcal{S}^{kma}$ is secure against random message attacks).*

Proof. Suppose there exists an adversary $\mathcal{A}$ who can break the full security of $\mathcal{S}$ in the standard security model. We construct a simulator $\mathcal{B}$ that breaks the security of $\mathcal{S}^{kma}$ in the KMA model. The interaction between the adversary $\mathcal{A}$ and the simulator $\mathcal{B}$ is described as follows.

Setup: The simulator $\mathcal{B}$ queries signatures to $\mathcal{S}^{kma}$ and receives a public key pk along with signed messages $(y_1, \sigma_{sk}[g^{y_1}]), \cdots, (y_{q_s}, \sigma_{sk}[g^{y_{q_s}}])$. Let $(g, h) \in \mathbb{G}$ be the public parameter of a secure chameleon hash function and τ be its

trapdoor key, such that $h = g^{\tau}$. The public key $PK = (pk, g, h)$ is forwarded to the adversary.

Query: The adversary queries the signature on the message m_i, which is adaptively chosen by itself. $\mathcal{B}$ computes $r_i = y_i - \tau m_i$ and compiles $\sigma_{sk}[y_i]$ into $\sigma^{\star}_{sk}[g^{y_i}]$. Then, $\mathcal{B}$ forwards the signature $(r_i, \sigma^{\star}_{sk}[g^{y_i}])$ to the adversary. It is a valid signature of m_i since $g^{y_i} = g^{\tau m_i + r_i} = h^{m_i} g^{r_i}$.

Output: The adversary outputs a forged signature $(m^*, r^*, \sigma^{\star}_{sk}[h^{m^*} g^{r^*}])$. We have that $y^* = \tau m^* + r^*$ is computable by $\mathcal{B}$. Then, $\mathcal{B}$ compiles the signature into $\sigma_{sk}[y^*]$ and outputs $(y^*, \sigma_{sk}[y^*])$ to break the $\mathcal{S}^{kma}$ scheme.

This completes the reduction. According to the security result in [19], when y_i are universally random, the chameleon hash is collusion-resistant. Therefore, the output of (m^*, r^*) satisfies $\tau m^* + r^* \neq \tau m_i + r_i$. I.E., $y^* = \tau m^* + r^*$ is a new message different from $y_1, y_2, \cdots, y_{q_s}$. We yield $\mathcal{B}$ breaks the security of $\mathcal{S}^{kma} = (\mathsf{KeyGen}^{\mathsf{kma}}, \mathsf{Sign}^{\mathsf{kma}}, \mathsf{Verify}^{\mathsf{kma}})$, and complete the proof. $\square$

We have completed the security transformation from known-message attacks to adaptive chosen-message attacks. The works in [19, 4] have already proposed a transformation from weakly (generic) chosen-message attacks to adaptive chosen-message attacks. In the weakly chosen-message model, the adversary can query signatures by selecting messages itself before receiving the public key. While the known-message model only allows the adversary to receive signed messages from the challenger. In comparison with the weakly chosen-message model, the known-message model restricts the adversary in selecting messages. From the adversary's view in message selection, our security transformation offers a more powerful security transformation, compared to the security transformation based on the weakly chosen-message model.

A similar security transformation from random message attacks to adaptive chosen-message attacks was proposed in [19]. Their approach can be applied to any signature scheme but it has to use a particular (strong) chameleon hash based on factoring assumptions. Our approach requires compilable signature schemes but is able to rely on a (weak) chameleon hash function based on discrete log assumptions. We stress that the core of our approach is similar to [19]. The compilable signature scheme enables us to make the weak chameleon hash capture the strong notion during the security transformation. The merit of our approach is to acquire smaller size of signatures since the representations of elements based on discrete log assumptions (e.g. 160 bits) is much smaller compared to that based on

factoring assumptions (e.g. 1024 bits) for the same security level.

5. Our Scheme

We present our signature construction in this section. The scheme will be reduced to the q-$\mathrm{SDH}_1^{\sqrt{q_B}}$ assumption, where q_B is the upper bound of signature numbers in generations and q is the smaller value of $\{q_s+1, \sqrt{q_B}+1\}$. We here only construct signatures secure against known-message attacks. Fully secure signatures can be realized using our security transformation in the Section 4.

5.1. Construction

Let the bilinear pairing be $(\mathbb{G}, \mathbb{G}_T, g, p, e)$. The message space in our scheme is $\mathbb{Z}_p$ and it can be naturally extended to arbitrary bit strings using a collusion-resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_p$. Our construction states as follows.

KeyGen: Randomly choose α, β from $\mathbb{Z}_p$, and set $g_1 = g^\alpha, g_2 = g^\beta$. Let q_B be the upper bound of signature generations, such that there exists an integer z satisfying $q_B = z^2$. The public key is $pk = (\mathbb{G}, \mathbb{G}_T, p, e, g, g_1, g_2, \sqrt{q_B})$ and the private key is $sk = (\alpha, \beta)$.

The stateful information S in our scheme is defined as $\mathsf{S} = (c_1, c_2, \gamma, g^\gamma, g^{\frac{\beta-H}{\alpha+c_1}})$.

- c_1, c_2 are two counters initialized with $c_1 := 1$ and $c_2 := 0$.
- γ is a variant initialized with a random value from $\mathbb{Z}_p$.
- $H = H(g^\gamma)$ and $g^{(\beta-H)/(\alpha+c_1)}$ is computed using α, β, H, c_1.

The private key sk and stateful information γ in S should be both secret.

Sign: Given a message m to be signed, the signer increases the counter c_2 by one as $c_2 := c_2 + 1$ and does as follows:

- If $c_2 \le \sqrt{q_B}$. Compute σ_5 and output the signature $\sigma_{sk}[m]$ as

$$\sigma_{sk}[m] = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5) = \left(c_1, g^{\frac{\beta-H}{\alpha+c_1}}, g^\gamma, c_2, g^{\frac{\gamma-m}{\alpha+c_2}}\right),$$

  where the elements $\sigma_1, \sigma_2, \sigma_3, \sigma_4$ are from the stateful information S.

- Otherwise $c_2 > \sqrt{q_B}$. Update the stateful information S and compute the signature $\sigma_{sk}[m]$ as

- Increase the counter c_1 by one as $c_1 := c_1 + 1$ and initialize c_2 with $c_2 := 1$.
- Randomly choose $\delta \in \mathbb{Z}_p$ and update $(\gamma, g^{\gamma}, g^{\frac{\beta-H}{\alpha+c_1}})$ with the updated c_1 and $\gamma = \delta$.
- If $c_1 > \sqrt{q_B}$, abort; otherwise compute σ_5 and output the signature $\sigma_{sk}[m]$ as

$$\sigma_{sk}[m] = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5) = \left(c_1, g^{\frac{\beta-H}{\alpha+c_1}}, g^{\gamma}, c_2, g^{\frac{\gamma-m}{\alpha+c_2}}\right),$$

  where $\sigma_1, \sigma_2, \sigma_3, \sigma_4$ are from the stateful information S.

Verify: Given the signed message $(m, \sigma_{sk}[m])$ and let the signature be $\sigma_{sk}[m] = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$. Compute $H = H(\sigma_3)$, and verify that $\sigma_1, \sigma_4 \leq \sqrt{q_B}$ and that

$$e(\sigma_2, g_1 g^{\sigma_1}) = e(g_2 g^{-H}, g),\ e(\sigma_5, g_1 g^{\sigma_4}) = e(\sigma_3 g^{-m}, g).$$

5.2. Main Features

In our signature scheme, the number q_B must be decided by the signer during the key generation phase. However, we say that this bound affects little on the flexibility of signing operations. We can choose a large enough q_B for each key pair, for example, $q_B = 2^{30}$ or $q_B = 2^{40}$ depending on applications.

The value q_B is used to constrain the flexibility of attacks. The inverted exponents $\alpha + c_1$ and $\alpha + c_2$ for valid signatures are restricted in the small range $[\alpha + 1, \alpha + \sqrt{q_B}]$. Only signatures with counter values no more than $\sqrt{q_B}$ will be accepted. That is, an adversary has to forge a valid signature $(\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$ satisfying $\sigma_1, \sigma_4 \leq \sqrt{q_B}$. We use this property to program security reduction to the q-$\mathrm{SDH}_1^{\sqrt{q_B}}$ problem.

Our public key is short, containing two group elements and the statement $\sqrt{q_B}$ besides the bilinear pairing. While the bilinear pairing can be shared by other users. We can utilize an asymmetric pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ to construct shortest possible signatures, defined in the section 2. For 80-bit security using asymmetric pairing, we have $|\sigma_2| = |\sigma_3| = |\sigma_5| = 160$; for $q_B = (2^{15} - 1)^2 \approx 2^{30}$, we have $|\sigma_1| = |\sigma_4| = 15$. Therefore, our signature can be 510 bits in length for the KMA security.

Regarding the efficiency of our signature scheme, each signature generation requires one exponentiation for $c_2 \leq \sqrt{q_B}$ and three for $c_2 > \sqrt{q_B}$. The verification requires four pairings and four exponentiations. We show

that the verification time can be halved using the "small exponent" idea for batch verification [1]. Our new verification time requires only two pairings and two multi-exponentiations. We will describe this efficient verification method later.

We improve the security of digital signature depending on the q-SDH$_1^{\sqrt{q_B}}$ assumption. The improved assumption is harder than the q_s-SDH assumption with twofold reasons. The value q is the smaller value of $\{q_s+1, \sqrt{q_B}+1\}$ and our q-SDH$_1^{\sqrt{q_B}}$ assumption are constrained in $\sqrt{q_B}$ correct answers only. However, the computation of our scheme is less efficient and it requires the stateful technique. While signatures based on the (q_s+1)-SDH problem can also be reduced to the q_s-SDH$_1^{\sqrt{q_B}}$ problem (still stronger than our q-SDH$_1^{\sqrt{q_B}}$ assumption), their security reductions are loose. As noted in [3], the loose reduction means that we have to grow the size of groups to compensate security loss. An implementation using a larger group size is less efficient in the group operations. This definitely slows down their signature generations and verifications.

5.3. Security

Theorem 2. *Our signature is (t, q_s, ϵ) KMA-secure assuming that the hash function H is collusion-resistant and the q-SDH$_1^{\sqrt{q_B}}$ assumption is (t', ϵ') secure.*

$$q = \begin{cases} q_s + 1 & q_s \leq \sqrt{q_B} \\ \sqrt{q_B} + 1 & \sqrt{q_B} < q_s \leq q_B \end{cases}, \ t' = t + O(q_s q t_e), \ \epsilon' \approx \epsilon$$

where t_e is the average of time cost for exponentiations in $\mathbb{G}$.

Proof. Suppose there exists an adversary $\mathcal{A}$ which (t, ϵ)-breaks our signature scheme though known-message attacks. We construct an algorithm $\mathcal{B}$ that solves the q-SDH$_1^{\sqrt{q_B}}$ assumption with advantage (t', ϵ'). The algorithm $\mathcal{B}$ is given the challenge input $(g, g^a, g^{a^2}, \cdots, g^{a^q}, \sqrt{q_B})$. The target is to output $(c, g^{1/(a+c)})$ for a freely chosen integer c such that $1 \leq c \leq \sqrt{q_B}$. The interaction between $\mathcal{A}$ and $\mathcal{B}$ is as follows.

Setup: Let γ_i be the variant for the stateful counter $c_1 := i$. $\mathcal{B}$ computes the public key as follows.

- $q_s \leq \sqrt{q_B}$. $\mathcal{B}$ randomly chooses a q-degree polynomial function $f_1(x) \in \mathbb{Z}_p[x]$ and sets $\gamma_1 = f_1(a)$. g^{γ_1} can be computed from the challenge input and $f_1(x)$. For the private key α, β, the algorithm $\mathcal{B}$ sets $\alpha = a$ and $\beta = f(a)$. Here, $f(x) \in \mathbb{Z}_p[x]$ is another 1-degree polynomial function satisfying $f(-1) = H(g^{\gamma_1})$. Then, $g^a, g^{f(a)}$ are also computable.

- $q_s > \sqrt{q_B}$. Suppose there exists an integer $2 \leq k \leq \sqrt{q_B}$ such that $(k\text{–}1)\sqrt{q_B} < q_s \leq k\sqrt{q_B}$. $\mathcal{B}$ randomly chooses k independent q-degree polynomial functions $f_1(x), f_2(x), \cdots, f_k(x) \in \mathbb{Z}_p[x]$ and sets $\gamma_i = f_i(a)$ for all $i = 1, 2, \cdots, k$. The element g^{γ_i} can be computed from the challenge input and $f_i(x)$. For the private key α, β, the algorithm $\mathcal{B}$ sets $\alpha = a$ and $\beta = f(a)$. Here, $f(x) \in \mathbb{Z}_p[x]$ is a k-degree polynomial function satisfying $f(-i) = H(g^{\gamma_i})$, which can be computed using the Lagrange-Interpolation approach. Then, $g^a, g^{f(a)}$ are also computable from the challenge input.

The simulator $\mathcal{B}$ forwards the public key pk to the adversary.

Query: The adversary queries the ith signature. Let the counter value be $(c_1, c_2) = (j_1, j_2)$. We have $j_1 = 1$ for $q_s \leq \sqrt{q_B}$ or $j_1 \leq k$ for $q_s > \sqrt{q_B}$. $\mathcal{B}$ lets the ith message to be signed be $m_i = f_{j_1}(-j_2)$. The corresponding signature is

$$\sigma_{sk}[m_i] = \left(j_1, g^{\frac{\beta - H}{\alpha + j_1}}, g^{\gamma_{j_1}}, j_2, g^{\frac{\gamma_{j_1} - m_i}{\alpha + j_2}}\right).$$

According to the setup phase, we have that

$$F_{j_1}(x) = \frac{f(x) - H(g^{\gamma_{j_1}})}{x + j_1}, \; F_{j_2}(x) = \frac{f_{j_1}(x) - f_{j_1}(-j_2)}{x + j_2}$$

are two (q–1)-degree polynomial functions. Then,

$$\begin{aligned} g^{\frac{\beta - H}{\alpha + j_1}} &= g^{\frac{f(a) - H}{a + j_1}} = g^{F_{j_1}(a)} \\ g^{\frac{\gamma_{j_1} - m_i}{\alpha + j_2}} &= g^{\frac{f_{j_1}(x) - f_{j_1}(-j_2)}{x + j_2}} = g^{F_{j_2}(a)} \end{aligned}$$

can be computed from the challenge input and $F_{j_1}(x), F_{j_2}(x)$. $\mathcal{B}$ forwards the signature to the adversary.

Output: The adversary outputs a signed message $(m^*, \sigma_{sk}[m^*])$. Let the forged signature $\sigma_{sk}[m^*]$ be

$$\sigma_{sk}[m^*] = (\sigma_1^*, \sigma_2^*, \sigma_3^*, \sigma_4^*, \sigma_5^*) = \left(c_1^*, g^{\frac{\beta - H^*}{\alpha + c_1^*}}, g^{\gamma^*}, c_2^*, g^{\frac{\gamma^* - m^*}{\alpha + c_2^*}}\right).$$

We have $1 \leq c_1^*, c_2^* \leq \sqrt{q_B}$; otherwise, it is an invalid signature. The reduction falls into two types associated with g^{γ^*}. Without loss of generality, we discuss the case $q_s > \sqrt{q_B}$ only.

- $g^{\gamma^*} \in \{g^{\gamma_1}, g^{\gamma_2}, \cdots, g^{\gamma_k}\}$. Without loss of generality, let $\gamma^* = \gamma_i$. If $x + c_2^* | f_i(x) - m^*$, abort; otherwise, we can rewrite σ_5^* into

$$g^{\frac{\gamma^* - m^*}{\alpha + c_2^*}} = g^{F(a) + \frac{d}{a + c_2^*}}$$

for some (q–1)-degree polynomial function $F(x)$ and some nonzero $d \in \mathbb{Z}_p$. Then, $\mathcal{B}$ outputs $(c_2^*, g^{1/(a+c_2^*)})$ as the solution to the q-SDH$_1^{\sqrt{q_B}}$ assumption.

- $g^{\gamma^*} \notin \{g^{\gamma_1}, g^{\gamma_2}, \cdots, g^{\gamma_k}\}$. If $x + c_1^* | f(x) - H^*$, abort; otherwise, we can rewrite σ_2^* into

$$g^{\frac{\beta - H^*}{\alpha + c_1^*}} = g^{F(a) + \frac{d}{a + c_1^*}}$$

for some (q–1)-degree polynomial function $F(x)$ and some nonzero $d \in \mathbb{Z}_p$. Then, $\mathcal{B}$ outputs $(c_1^*, g^{1/(a+c_1^*)})$ as the solution to the q-SDH$_1^{\sqrt{q_B}}$ assumption.

This completes the description of our reduction. We now analyze the successful probability ϵ' and the time complexity t'. The reduction aborts only when $x + c_2^* | f_i(x) - m^*$ or $x + c_1^* | f(x) - H^*$ holds. They are analyzed as following.

- $g^{\gamma^*} \in \{g^{\gamma_1}, g^{\gamma_2}, \cdots, g^{\gamma_k}\}$ and let $\gamma^* = \gamma_i$. It falls into two cases.
  - $\mathcal{B}$ ever used the counter values (i, c_2^*) to compute a signature $\sigma_{sk}[m']$ on m' in the query phase. Let $\sigma_{sk}[m'] = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$ and we have

$$(\sigma_3, \sigma_4, \sigma_5) = \Big(g^{\gamma_i}, c_2^*, g^{\frac{\gamma_i - m'}{\alpha + c_2^*}}\Big).$$

    Since $m' \neq m^*$ and $x + c_2^* | f_i(x) - m'$ in simulation, we deduce that

$$gcd\Big(x + c_2^*, f_i(x) - m^*\Big) = 1$$

    holds with probability 1.
  - $\mathcal{B}$ never used the counter values (i, c_2^*) to compute a signature in the query phase. We have that $x + c_2^* | f_i(x) - m^*$ holds when $m^* = f_i(-c_2^*)$. However, $f_i(-c_2^*)$ is universally random and independent in $\mathbb{Z}_p$ since $f_i(x)$ is a q-degree polynomial function. Hence, $m^* = f_i(-c_2^*)$ holds with probability $1/p$ at most.

- $g^{\gamma^*} \notin \{g^{\gamma_1}, g^{\gamma_2}, \cdots, g^{\gamma_q}\}$. It falls into two cases.
  - $\mathcal{B}$ ever used the counter values c_1^* to compute a signature $\sigma_{sk}[m']$ on m' in the query phase. Let $\sigma_{sk}[m'] = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$ and we have
  $$(\sigma_1, \sigma_2, \sigma_3) = \Big(c_1^*, g^{\frac{\beta - H}{\alpha + c_1^*}}, g^{\gamma_i}\Big).$$
  Since $H(g^{\gamma_i}) \neq H(g^{\gamma^*})$ and $x + c_1^* | f(x) - H$ in simulation, we deduce that
  $$gcd\Big(x + c_1^*, f(x) - H^*\Big) = 1$$
  holds with probability 1.
  - $\mathcal{B}$ never used the counter values c_1^* to compute a signature in the query phase. We have that $x + c_1^* | f(x) - H^*$ holds when $H^* = f(-c_1^*)$. However, $f(-c_1^*)$ is universally random and independent in $\mathbb{Z}_p$ since $f(x)$ is a k-degree polynomial function. Hence, $H^* = H(g^{\gamma^*}) = f_i(-c_2^*)$ holds with probability $1/p$ at most.

Therefore, the probability of successful reduction is $1 - 1/p$ at least. If the adversary can output a valid signature with probability ϵ, $\mathcal{B}$ can use the forgery to solve the q-$\mathrm{SDH}_1^{\sqrt{q_B}}$ assumption with probability $\epsilon' \approx \epsilon$. The time complexity for $\mathcal{B}$ is mainly dominated by signature generations in the query phase. For each signature computation, it takes roughly $O(q)$ exponentiations. Thus, we have $t' = t + O(q_s q t_e)$ and $\epsilon' \approx \epsilon$, where t_e is the exponentiation time. This completes our reduction proof. □

Theorem 3. *A fully secure signature can be constructed from our KMA secure signature scheme.*

Proof. According to the theorem 1 and theorem 2, we only need to prove that our signature is a compilable signature and signed messages are universally random.

Let $g \in pk$ be the generator for chameleon hash. It is easy to verify that our construction is a compilable signature scheme. We have

$$\begin{aligned} \sigma_{sk}[m] &= \Big(c_1, g^{\frac{\beta - H}{\alpha + c_1}}, g^{\gamma}, c_2, g^{\frac{\gamma - m}{\alpha + c_2}}\Big) \\ \sigma_{sk}^{\star}[g^m] &= \Big(c_1, g^{\frac{\beta - H}{\alpha + c_1}}, g^{\gamma}, c_2, g^{\frac{\gamma}{\alpha + c_2}} \cdot (g^m)^{\frac{-1}{\alpha + c_2}}\Big) \end{aligned}$$

and they can be converted each other. The algorithm $\mathsf{Verify}^{\star}$ is defined as follows.

Verify*: Given the signed message $(y, \sigma^\star_{sk}[y])$ and let the signature be $\sigma^\star_{sk}[y] = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$. Compute $H = H(\sigma_3)$, and verify that $\sigma_1, \sigma_4 \leq \sqrt{q_B}$ and that

$$e(\sigma_2, g_1 g^{\sigma_1}) = e(g_2 g^{-H}, g),\ e(\sigma_5, g_1 g^{\sigma_4}) = e(\sigma_3 y^{-1}, g).$$

Signed messages are universally random. According to the security proof in the theorem 2, the signed message responded in the query phase is defined as $f_{j_1}(-j_2)$. Since $f_i(x) \in \mathbb{Z}_p$ for all $i = 1, 2, \cdots, k$ are q-degree random polynomial functions, $f_{j_1}(-j_2)$ for $j_2 \in \{1, 2, \cdots, \sqrt{q_B}\}$ are universally random and independent in $\mathbb{Z}_p$. This says that all signed messages in the query phase are universally random in $\mathbb{Z}_p$ and we complete the proof. □

5.4. Online/Offline Signature

The notion of online/offline signature [10] was introduced for efficient signing computations. The signing operation is separated into two phases: offline and online. In the offline phase, most of heavy computations are pre-computed before a message is given. In the online phase, only very light computation is required after a message to be signed is given.

Our signature can also be modified into online/offline signatures. The signature on m of full security is defined as

$$\sigma_{sk}[m] = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6) = \left(c_1, g^{\frac{\beta - H}{\alpha + c_1}}, g^\gamma, c_2, g^{\frac{\gamma - (\tau m + r)}{\alpha + c_2}}, r\right),$$

where $\tau \in \mathbb{Z}_p$ is the trapdoor key of chameleon hash. It can be changed into online/offline signing with the same idea as in [4]. In the offline phase, the signer updates its stateful information, chooses at random $r' \in \mathbb{Z}_p$ and computes $\sigma_5 = g^{r'}$. The parameters $(\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, r')$ are stored in the offline phase. In the online phase, given a message m to be signed, the signer only does the following modular computation

$$r = \gamma - r'(\alpha + c_2) - \tau m = \sigma_6 \pmod p,$$

and outputs the signature $\sigma_{sk}[m] = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6)$. It is easy to verify its correctness where we have

$$g^{r'} = g^{\frac{\gamma - \tau m - r}{\alpha + c_2}} = g^{\frac{\gamma - (\tau m + r)}{\alpha + c_2}}.$$

5.5. Verification Improvement

The original verification algorithm defined in the subsection 5.1 requires four pairings and four exponentiations. We redefine an algorithm Verify to speed up the signature verification. We use small exponent [1] to combine two separated computations together. The new verification algorithm for fully secure scheme is defined as following.

Verify: Given the signed message $(m, \sigma_{sk}[m])$ and let the signature be

$$\sigma_{sk}[m] = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6) = \Big(c_1, g^{\frac{\beta - H}{\alpha + c_1}}, g^{\gamma}, c_2, g^{\frac{\gamma - (\tau m + r)}{\alpha + c_2}}, r\Big).$$

The verifier does as follows to verify the signature.

- Compute $H = H(\sigma_3)$, and verify that $\sigma_1, \sigma_4 \leq \sqrt{q_B}$;
- Randomly choose two l-bit strings t_1, t_2 and verify that

$$e\Big(\sigma_2^{t_1}\sigma_5^{t_2}, g_1\Big) = e\Big(g_2^{t_1}\sigma_3^{t_2}h^{-t_2 m}g^{-(Ht_1 + mt_2)}\sigma_2^{-\sigma_1 t_1}\sigma_5^{-\sigma_4 t_2}, g\Big).$$

For the correctness of this verification, we have the following theorem.

Theorem 4. *The verifier accepts an invalid signature with probability* $1/2^l$.

Proof. Let $\sigma_{sk}^f[m]$ be a fake signature on the message m. We can rewrite the fake signature into

$$\sigma_{sk}^f[m] = \Big(c_1, g^{\frac{\beta - H}{\alpha + c_1} + \eta_1}, g^{\gamma}, c_2, g^{\frac{\gamma - (\tau m + r)}{\alpha + c_2} + \eta_2}, r\Big) = (\sigma_1, \sigma_2 g^{\eta_1}, \sigma_3, \sigma_4, \sigma_5 g^{\eta_2}, \sigma_6)$$

for some unknown $\eta_1, \eta_2 \in \mathbb{Z}_p$. One of them must be nonzero; otherwise, it is a valid signature. Without loss of generality, suppose $\eta_1 \neq 0$. If this fake signature passes the signature verification as above, we deduce that

$$e\Big(g^{t_1\eta_1 + t_2\eta_2}, g^{\alpha}\Big) = e\Big(g^{-(\sigma_1 t_1 \eta_1 + \sigma_4 t_2 \eta_2)}, g\Big)$$

and that

$$t_1 = -\frac{\eta_2(\alpha + \sigma_4)}{\eta_1(\alpha + \sigma_1)} \cdot t_2.$$

Since t_1 is an l-bit string universally random and independent from $\{0,1\}^l$, we have that t_1 happens to be equal to $-\eta_2(\alpha + \sigma_4)t_2/(\eta_1\alpha + \eta_1\sigma_1)$ with probability $1/2^l$. Therefore, an invalid signature is accepted with $1/2^l$ at most. We can set $l = 40$ in choice such that the probability of accepting an invalid signature is negligible. This completes our proof. □

6. Conclusion

The q-SDH assumption is not a desirable assumption because there are exponential answers for a given challenge input. In this paper, we defined the q-SDH_1^R assumption, which requires only polynomial R correct answers. The q-SDH_1^R assumption with only R answers is much more reliable than the q-SDH assumption (or, q-SDH_1^p assumption), because R is much smaller than p. We proposed a new signature scheme whose security can be tightly reduced to the q-SDH_1^R assumption without random oracles. The value R is only $\sqrt{q_B}$, where q_B is the upper bound of the number of signatures, and q is the smaller value of $\{q_s + 1, \sqrt{q_B} + 1\}$. We also presented a new security transformation of fully secure signatures from weakly secure signature against known-messages attacks. Although the transformation is conditional, it provides a new way to construct fully secure signatures.

References

[1] M.Bellare, J.Garay, T.Rabin, Fast batch verification for modular exponentiation and digital signatures, in: EUROCRYPT 1998, Lecture Notes in Computer Science, vol. 1403, Springer, Heidelberg, 1998, pp. 236-250.

[2] M.Bellare, P.Rogaway, Random oracles are practical: A paradigm for designing efficient protocols, in: ACM CCS, ACM Press, NewYork, USA, 1993, pp. 62-73.

[3] M.Bellare, T.Ristenpart, Simulation without the Artificial Abort: Simplified Proof and Improved Concrete Security for Waters' IBE Scheme, in: EUROCRYPT 2009, Lecture Notes in Computer Science, vol. 5479, Springer, Heidelberg, 2009, pp.407-424.

[4] D.Boneh, X.Boyen, Short signatures without random oracles, in: EUROCRYPT 2004, Lecture Notes in Computer Science, vol. 3027, Springer, Heidelberg, 2004, pp. 56-73.

[5] D.Boneh, X.Boyen, Short signatures without random oracles, J. Cryptology, 21(2), (2008), pp. 149-177.

[6] D.Boneh, B.Lynn, H.Shacham, Short signatures from the Weil pairing, in: ASIACRYPT 2001, Lecture Notes in Computer Science, vol. 2248, Springer, Heidelberg, 2001, pp. 514-532.

[7] R.Canetti, O.Goldreich, S. Halevi, The random oracle methodology, revisited, in: STOC 1998,ACM Press, 1998, pp. 209-218.

[8] J.H.Cheon, Security analysis of the strong diffie-hellman problem, in: EUROCRYPT 2006, Lecture Notes in Computer Science, vol. 4004. Springer, Heidelberg, 2006, pp. 1-11.

[9] R. Cramer,V. Shoup, Signature schemes based on the strong RSA assumption, in: ACM CCS, ACM Press, 1999, pp. 46-51.

[10] S.Even, O.Goldreich, S.Micali, Online/offline digital signatures, in: CRYPTO 1989, Lecture Notes in Computer Science, vol. 435, Springer, Heidelberg, 1990, pp. 263-275.

[11] R.Gennaro, S.Halevi,T.Rabin, Secure hash-and-sign signatures without the random oracle, in: EUROCRYPT 1999, Lecture Notes in Computer Science, Springer, Heidelberg, 1999, pp. 123-139.

[12] C.Gentry, Practical identity-based encryption without random oracles, in: EUROCRYPT 2006, Lecture Notes in Computer Science, vol.4004, Springer, Heidelberg, 2006, pp. 445-464.

[13] E.J.Goh, S.Jarecki, A signature scheme as secure as the Diffie- Hellman problem, in: EUROCRYPT 2003, Lecture Notes in Computer Science, Springer, Heidelberg, 2003, pp. 401-415.

[14] S. Goldwasser, S.Micali, R. Rivest,A digital signature scheme secure against adaptive chosen-message attacks, SIAM J. Comput. 17 (2) (1988) 281-308.

[15] D.Hofheinz, E.Kiltz, Programmable hash functions and their applications, in: CRYPTO 2008. Lecture Notes in Computer Science, vol. 5157, Springer, Heidelberg, 2008,pp. 21-38.

[16] S.Hohenberger, B.Waters, Realizing hash-and-sign signatures under Standard Assumptions, in: EUROCRYPT 2009, Lecture Notes in Computer Science, vol. 5479, Springer, Heidelberg, 2009, pp. 333-350,

[17] S. Hohenberger, B.Waters, Short and stateless signatures from the RSA Assumption, in: CRYPTO 2009, Lecture Notes in Computer Science, vol.5677, Springer, Heidelberg, 2009, pp.654-670.

[18] H.Krawczyk, T.Rabin, Chameleon signatures, in: NDSS 2000, Internet Society, 2000, pp 143-154.

[19] A.Shamir, Y.Tauman, Improved online/offline signature schemes, In: CRYPTO 2001, Lecture Notes in Computer Science, vol. 2139, Springer, Heidelberg, 2001, pp.355-367.

[20] B.Waters, Efficient identity-based encryption without random oracles, In: EUROCRYPT 2005, Lecture Notes in Computer Science, vol. 3494, Springer, Heidelberg, 2005, pp. 320-329.