Information Sciences 201 (2012) 128–139

Contents lists available at SciVerse ScienceDirect

Information Sciences

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

Generic security-amplifying methods of ordinary digital signatures ☆

Jin Li [a,*], Fangguo Zhang [b], Xiaofeng Chen [c], Kwangjo Kim [d], Duncan S. Wong [e]

[a] School of Computer Science and Educational Software, Guangzhou University, Guangzhou 510006, PR China
[b] School of Information Science and Technology, Sun Yat-Sen University, Guangzhou 510275, PR China
[c] Key Laboratory of Computer Networks and Information Security, Xidian University, PR China
[d] Department of Computer Science, Korea Advanced Institute of Science and Technology, 119 Munjiro, Yusong-ku, Daejeon 305-714, Republic of Korea
[e] Department of Computer Science, City University of Hong Kong, Hong Kong, PR China

ARTICLE INFO

Article history:
Received 1 June 2010
Received in revised form 6 March 2012
Accepted 9 March 2012
Available online 17 March 2012

Keywords:
Signature
Weak chosen message attack
q-SDH assumption
Strong-RSA assumption
One-time signature
Strong unforgeability

ABSTRACT

Digital signatures are one of the most fundamental primitives in cryptography. In this paper, three new paradigms are proposed to obtain signatures that are secure against existential forgery under adaptively chosen message attacks (*fully-secure*, in short), from any *weakly-secure* signature. These transformations are generic, simple, and provably secure in the standard model. In the first paradigm, based on a *weakly-secure* signature scheme, the construction of a *fully-secure* signature scheme requires one-time signature additionally. However, the other two are built only on *weakly-secure* signatures. To the best of our knowledge, it is observed for the first time in this paper that two *weakly-secure* signature schemes are sufficient to construct a *fully-secure* signature scheme.

Based on the new proposed paradigms, several efficient instantiations without random oracles are also presented. We also show that these *fully-secure* signature schemes have many special interesting properties in application.

1. Introduction

Digital signature plays a central role in cryptography. The standard definition on the security of signature scheme was given by Goldwasser et al. [18]. Compared to the standard security model [18], there are also many weak security models. In fact, in terms of the goals and resources of the adversary, many security models can be formed. However, signatures in these weak security models, such as existentially unforgeable against generic chosen message attack (or, weak chosen message attack) [5,18], are not sufficient in many practical applications. In this paper, the signatures that is secure against weak chosen message attack are called *weakly-secure* signatures. In this security model, the adversary is required to submit all signature queries before the signer's public key is published.

Obviously, because of the limitation of signature queries, the signature that is provably secure in this weak model will be insecure in many practical applications. There are many attempts to design practical and provably secure signature schemes in the standard security model [18]. These methods can be divided into two categories, namely, concrete construction method and generic construction method.

There are many concrete constructions of signature schemes based on some standard assumptions, such as discrete logarithm problem [28,30], computational Diffie–Hellman problem [6,17,34], factoring problem [3]. Some constructions based on other assumptions [29,36] have also been proposed. Though they are efficient, their security can only be proven in the

☆ An extended abstract of this paper has appeared in the proceedings of the ACNS'08 [24].

* Corresponding author.
E-mail address: jinli71@gmail.com (J. Li).

0020-0255/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2012.03.006

random oracle model. As we know, Canetti et al. [9] has showed that some popular cryptosystems previously proved secure in the random oracle are actually insecure when the random oracle is instantiated by any real-world hash functions. Over the last decades, several signature schemes were proposed in the standard model based on some stronger complexity assumptions such as [5,8,12,16,20]. Among them, the most efficient schemes are based on the Strong-RSA assumption [12,16] and q-strong Diffie–Hellman (q-SDH) assumption [5], which are cryptographically stronger than the computational Diffie–Hellman and RSA assumptions.

There are also many generic constructions of signatures based on the basic cryptographic primitive, such as (trapdoor) one-way functions [1,22]. Many generic constructions from other cryptographic protocols have also been proposed, such as non-interactive zero-knowledge [19,11,15]. Among them, the most famous one is the Fiat–Shamir (FS) transform [15]. However, its security relies on the random oracle model. To avoid the usage of random oracle model, from the Σ protocol, Cramer and Damgård [11] gave another generic transformation. However, this conversion method is not practical because it used the authentication tree. Very recently, Bellare and Shoup [4] showed a simple transform for the construction of standard and strongly secure signatures from the Σ protocol, using the tool of two-tier signatures.

1.1. Our results

Firstly, we present three new paradigms to transform any *weakly-secure* signature schemes into *fully-secure* signature schemes. More precisely, these three paradigms are called sequential composition with one-time signature, sequential composition method, and parallel composition method, respectively. To the best of our knowledge, it is observed for the first time in our paper that only two *weakly-secure* signature schemes are required to get *fully-secure* signatures. Therefore, this paper makes contributions towards the goal to obtain efficient constructions with standard assumptions. Motivated from the results of [23], Huang et al. [21] showed how to construct a strongly unforgeable signature from a weakly secure signature and Li et al. [24] showed two generic construction methods to get an unforgeable signature scheme from a *weakly-secure* signature scheme. Thus, these results have interest from both theoretical and practical perspective. More specifically, these three paradigms are described as follows:

- **Sequential composition with one-time signature**: This paradigm requires one weak signature scheme and an ordinary one-time signature. Key pair of the weak signature scheme is generated in key generation algorithm and used to sign the public key of one-time signature, which is generated in signing algorithm.
- **Sequential composition** (of weak signatures): This paradigm requires two weak signature schemes sequentially. Key pair in the first weak signature scheme is generated in key generation algorithm. During signing algorithm, another key pair of weak signature is generated. The first secret key is used to sign the other public key, and the other secret key is used to sign a message.
- **Parallel composition** (of weak signatures): Two weak signature schemes are also required in this paradigm, however, both of their key pairs should be generated in key generation algorithm, and used to sign two random and related messages.

We also show several efficient instantiations without random oracles converted from two *weakly-secure* signature schemes. The first two paradigm are very efficient in key generation compared to the last. However, the signing algorithm of the last paradigm is more efficient. There is a coincidence that, when instantialized from weak signature scheme [16], the construction will be similar to twin signature scheme [26]. In fact, the last paradigm can be viewed as generalization and extension of the twin signature scheme [26]. Recently, another notion called strongly existential unforgeability was concerned by many contributions, such as [7,21,33].

1.2. Organization

In Section 2, the definitions of variant signatures are given. Then, two previous instantiations of *weakly-secure* signature schemes are reviewed in Section 3. In Section 4, three generic transformations techniques are proposed. In Section 5, several instantiations from sequential composition with one-time signature scheme are presented. In Section 6, instantiations from sequential composition method are given. In Section 7, we show the two instantiations from parallel composition method. The conclusion is given in Section 8.

2. Preliminaries

A signature scheme is defined by the following algorithms:

- *Key generation algorithm* Gen. On input 1^k, where k is the security parameter, it outputs (pk, sk) as public and secret keys.
- *Signing algorithm* Sign. On input a message m and sk, it outputs a signature σ.
- *Verification algorithm* Verify. Given public key pk, message m and signature σ, algorithm $\mathsf{Verify}(pk, m, \sigma)$ outputs 1 if $\sigma \leftarrow \mathsf{Sign}(sk, m)$. Otherwise, output 0.

In terms of adversary's goals, it can be divided into four categories: (1) *Total break*: This is the most serious attack, in which the adversary is able to disclose the secret key of the signer; (2) *Universal forgery*: The adversary is able to sign any given messages; (3) *Existential forgery*: The adversary is able to provide a signature on a new message whose signature has not been seen; (4) *Strong existential forgery*: The adversary is able to provide a new message-signature pair.

On the other hand, various resource can be made available to the adversary, helping into his/her forgery. We focus ourselves on two kinds of message attacks: (1) *Weakly chosen message attack*: The messages chosen by the adversary must be given to the signer before seeing the signer's public key; (2) *Adaptively chosen message attack*: The adversary is allowed to request signatures of messages chosen by itself.

2.1. Unforgeability

The standard notion of security for a signature scheme is called existential unforgeability under adaptively chosen message attacks (*fully-secure* signatures) [18], which is also required in other signature notions such as proxy signature and ring signature [13,31,35]. It can be defined through the following game:

Setup: A public/private key pair $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$ 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, that is, for each $i,\ 1 \leqslant i \leqslant q,\ \mathcal{A}$ 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 (*i.e.*, $\sigma_i = \mathsf{Sign}(sk, m_i)$).
Forge: $\mathcal{A}$ outputs a forgery (m^*, σ^*) and halts. $\mathcal{A}$ wins if

- σ^* is a valid signature on message m^* under the public key pk, i.e., $\mathsf{Verify}(pk, m^*, \sigma^*) = 1$; and
- m^* has never been queried, *i.e.*, $m^* \notin \{m_1, m_2, \ldots, m_q\}$.

Definition 1 (*Unforgeability*). A signature scheme $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is (t, q, ε)-*fully-secure*, if any adversary with run-time t wins the above game with probability at most ε after issuing at most q signing queries.

2.2. Strong existential unforgeability

The notion is also defined using the above game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$, except the definition that "$\mathcal{A}$ wins the game" is $\mathcal{A}$ can output a pair (m^*, σ^*) such that (m^*, σ^*) does not belong to the previous queried set $\{(m_i, \sigma_i)\}$ and $\mathsf{Verify}(pk, m^*, \sigma^*) = 1$.

2.3. Weak unforgeability

If we lower the adversary's ability to weak chosen message attack while keeping the goal of the adversary unchanged, we can get a weaker definition compared to existential unforgeability against adaptive chosen message attacks. The difference between this security notion with the standard security is that here it requires that the adversary should submit all messages for signature queries before the public key is seen. It is defined through the following game:

Pre-Proceeding: Adversary $\mathcal{A}$ runs for time t and issues q signing queries to a signing oracle, i.e., $\mathcal{A}$ chooses messages m_i, where $1 \leqslant i \leqslant q$.
Setup: A public/private key pair $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$ is generated and adversary $\mathcal{A}$ is given the public key pk. Meanwhile, q signatures σ_i on m_i from the signing oracle (*i.e.*, $\sigma_i = \mathsf{Sign}(sk, m_i)$), are also returned to $\mathcal{A}$.
Forge: $\mathcal{A}$ outputs a forgery (m^*, σ^*) and halts. $\mathcal{A}$ wins if

- σ^* is a valid signature on message m^* under the public key pk, i.e., $\mathsf{Verify}(pk, m^*, \sigma^*) = 1$; and
- m^* has never been queried, *i.e.*, $m^* \notin \{m_1, m_2, \ldots, m_q\}$.

And we define "$\mathcal{A}$ wins the game" is equivalent to $\mathcal{A}$ can output a pair (m^*, σ^*) such that σ is a valid signature of a new message m^*.

Definition 2 (*Weak Unforgeability*). A signature scheme $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ is (t, q, ε)-*weakly-secure*, if any adversary with run-time t wins the above game with probability at most ε.

3. Instantiations of weak signatures

It has been shown in [5,16] that two *weakly-secure* signature schemes can be constructed, based on the q-SDH assumption and Strong-RSA assumption, respectively, in the standard model.

3.1. Weak Boneh–Boyen signature [5]

Before describing the weak Boneh–Boyen signature, we first introduce some preliminaries on bilinear maps and an assumption used in [5].

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be cyclic groups of prime order p with the multiplicative group action. And, g is a generator of $\mathbb{G}_1$. Let $\hat{e}: \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a map with the following properties, (1) Bilinearity: $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$ for all $g_1, g_2 \in \mathbb{G}_1$, and $a, b \in_R \mathbb{Z}_p$; (2) Non-degeneracy: There exists $g_1, g_2 \in \mathbb{G}_1$ such that $\hat{e}(g_1, g_2) \neq 1$, in other words, the map does not send all pairs in $\mathbb{G}_1 \times \mathbb{G}_1$ to the identity in $\mathbb{G}_2$; (3) Computability: There is an efficient algorithm to compute $\hat{e}(g_1, g_2)$ for all $g_1, g_2 \in \mathbb{G}_1$.

As shown in [6,36], such non-degenerate bilinear maps over cyclic groups can be obtained from the Weil or the Tate pairing over algebraic curves.

Definition 3 (*q-Strong Diffie–Hellman Assumption (q-SDH in short)*). The q-SDH assumption in group $\mathbb{G}_1$ is defined as follows: given a $(q+1)$-tuple $(g, g^x, g^{x^2}, \cdots, g^{x^q}) \in (\mathbb{G}_1)^{q+1}$ as input, it is hard to output a pair $(c, g^{1/(x+c)})$, where $c \in \mathbb{Z}_p^*$.

Next, we describe the weak Boneh–Boyen signature [5].

1. **Gen:** Pick a random value $x \in \mathbb{Z}_p^*$ and compute $y = g^x$. Then, output the key pair (x, y), where y is the public key and x is the secret key.
2. **Sign:** Given a message $m \in \mathbb{Z}_p^*$, the signer computes the signature as $\sigma = g^{\frac{1}{x+m}}$ with his secret key x.
3. **Verify:** To verify the signature σ on message m, it checks if the following equation holds: $\hat{e}(y \cdot g^m, \sigma) = \hat{e}(g, g)$. If it holds, the signature is valid. Otherwise, the signature is invalid.

Theorem 1. *The weak Boneh–Boyen signature is weakly-secure if the q-SDH assumption holds.*

Proof. Refer to [5]. □

3.2. Weak GHR signature [16]

Gennaro et al. proposed a secure signature scheme [16] (denoted by GHR signature) without random oracle, however, under the assumption that hash function H is division intractable, and acts like the random oracle model or achieves the chameleon property, which was called a non-standard randomness-finding oracle in [16]. Division intractability means that it is computationally impossible to find $a_1, a_2, \ldots, a_k$ and b such that $H(b)$ divides the product of all the $H(a_i)$. In order to get a *fully-secure* signature without random oracles, the non-standard randomness-finding oracle was required [16]. This non-standard assumption helps the simulator to find the second preimage during the simulation. The randomness-finding oracle is non-standard because it requires that, given a hash function H, values M and e, one could find a random value R such that $H(R, M) = e$. In fact, without the assumption of randomness-finding oracle, the simulator has to guess which messages the adversary will ask during the signing simulation phase. This problem was also addressed in [10].

Definition 4 (*Strong-RSA Assumption*). Given a randomly chosen RSA modulus n, and a random element $s \in \mathbb{Z}_n^*$, it is infeasible to find a pair (e, r) with $e > 1$ such that $r^e = s \pmod{n}$.

We describe the weak GHR signature scheme as follows:

1. **Gen:** Pick two random safe primes p and q and compute $n = pq$ as RSA modulus. A hash function H and a random value $s \in \mathbb{Z}_n^*$ are selected. The public key is (n, s) and the secret key is (p, q).
2. **Sign:** To sign a message m, the signer computes $e \leftarrow H(m)$ and outputs the signature as $\sigma = s^{\frac{1}{e}} \bmod n$.
3. **Verify:** On input verification key (n, s), message m, and σ, check if the following equation $\sigma^{H(m)} = s \bmod n$. If it holds, output 1 and accept the signature. Otherwise, output 0.

Theorem 2. *The weak GHR signature scheme is weakly-secure if the Strong-RSA assumption holds and H is division intractability.*

Proof. Refer to [10,16] □

Note that the division-intractable hash functions can be constructed from collision-intractable hash functions [27].

4. Fully-secure signatures from weakly-secure signatures

There are two main techniques to get *fully-secure* signatures from *weakly-secure* signatures in literature, (1) Random Oracle Model: By using the hash function on the messages for signatures without changing other algorithms, the new signatures

can be *fully-secure* from the back patch property of random oracle [2]. This method was used in [5,36]; (2) Chameleon Hash Function: By combining *weakly-secure* signatures with the chameleon 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. Many papers have used this technique, such as [5,14,25,32].

In this section, three new paradigms are proposed to transform any *weakly-secure* signature into *fully-secure* signature.

4.1. Sequential composition method with one-time signature

Given a *weakly-secure* signature scheme $\Pi' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Verify}')$, we construct *fully-secure* signature scheme $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$. In the construction, we use a one-time signature scheme $\mathcal{OTS} = (\mathsf{OGen}, \mathsf{OSign}, \mathsf{OVerify})$, where OGen, OSign, and OVerify are generating keys algorithm, signing algorithm, and verifying signatures algorithm, respectively.

The construction of Π proceeds as follows:

1. **Gen.** On input security parameter 1^k, invoke $\mathsf{Gen}'(1^k)$ and obtain $(pk, sk) \leftarrow \mathsf{Gen}'(1^k)$. Output Π's public key pk and secret key sk (In fact, $\mathsf{Gen} = \mathsf{Gen}'$).
2. **Sign.** To sign message m, the signer first invokes $\mathsf{OGen}(1^k)$ to obtain one-time signature key pair $(opk, osk) \leftarrow \mathsf{OGen}(1^k)$. The signer then invokes algorithms $\mathsf{Sign}'(sk, opk)$ and $\mathsf{OSign}(osk, m)$. Output $\sigma = (A, B, C)$ as the signature, where $A = \mathsf{Sign}'(sk, opk)$, $B = \mathsf{OSign}(osk, m)$, $C = opk$.
3. **Verify.** On input verifying key pk, message m, and $\sigma = (A, B, C)$, output 1 if and only if $\mathsf{Verify}'(pk, A, C) = 1$ and $\mathsf{OVerify}(C, m, B) = 1$.

Key generation of the resulted *fully-secure* signature is the same with the key generation of weak signature. So, length of the public key in the *fully-secure* signature is independent of the one-time signature's public key length. In signature generation phase, $\mathsf{Sign}'(sk, opk)$ can be pre-computed by the signer. So, online computation in signature generation are only the computation of the one-time signature $\mathcal{OTS}$, which is very efficient. The computation in verification is just the verification computation of Π' and $\mathcal{OTS}$.

We first give some intuition as to why Π is secure against adaptively chosen message attack. Given only *weakly-secure* signature S' and one-time signature $\mathcal{OTS}$, the simulator can answer the adaptively signature queries from adversary because the choose of one-time keys is independent of messages chosen by the adversary, which implies that the one-time public keys can send to S' for signatures before messages are given, and then using $\mathcal{OTS}$ to sign messages from the adversary. Let $\sigma_i = (A_i, B_i, C_i)$ be the queried signatures and let $\sigma^* = (A^*, B^*, C^*)$ be the forged signature on a new message m^* outputted by the adversary. On one hand, if $C^* \neq C_i$ for $i = 1, \ldots, q_S$, then it implies that the Π' is insecure under weak chosen message attack. On the other hand, if $C^* = C_i$ for some signature output by the simulator, then B^* is another valid signature with respect to the one-time key C^*, that is to say, the adversary breaks the one-time signature scheme. So, under the assumption that Π' is *weakly-secure* and that $\mathcal{OTS}$ is secure one-time signature, the signature scheme Π is *fully-secure*.

Below, we formally prove the security of the signature scheme Π.

Theorem 3. *If Π' is a weakly-secure signature scheme and $\mathcal{OTS}$ is an unforgeable one-time signature scheme, then Π is a fully-secure signature scheme.*

Proof. See Appendix A. □

4.2. Sequential composition method

Given a *weakly-secure* signature scheme $\Pi' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Verify}')$, we construct a *fully-secure* signature scheme $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ by using the sequential composition method. We assume that the public key space belongs to the message space in this paradigm. Otherwise, hash function or other techniques could be applied here to achieve this. The construction of Π proceeds as follows:

- **Gen.** On input security parameter 1^k, invoke $\mathsf{Gen}'(1^k)$ and obtain $(pk, sk) \leftarrow \mathsf{Gen}'(1^k)$. Output Π's public key pk and secret key sk (In fact, $\mathsf{Gen} = \mathsf{Gen}'$).
- **Sign.** To sign message m, the signer first invokes $\mathsf{Gen}'(1^k)$ to obtain a key pair $(pk', sk') \leftarrow \mathsf{Gen}'(1^k)$. The signer then invokes algorithms $\mathsf{Sign}'(sk, pk')$ and $\mathsf{Sign}'(sk', m)$. Finally, it outputs $\sigma = (A, B, C)$ as the signature, where $A = \mathsf{Sign}'(sk, pk')$, $B = \mathsf{Sign}'(sk', m)$, $C = pk'$.
- **Verify.** On input verifying key pk, message m, and signature $\sigma = (A, B, C)$, output 1 if and only if $\mathsf{Verify}'(pk, C, A) = 1$ and $\mathsf{Verify}'(C, m, B) = 1$.

Key generation of the resulted *fully-secure* signature Π is the same with the key generation of weak signature Π'. In signature generation phase, $\mathsf{Sign}'(sk, pk')$ can be pre-computed by the signer. The construction is similar with [4,23]. However, it

is observed for the first time that only *weakly-secure* signatures are required here, instead of *fully-secure* signature scheme [4] or one-time signature scheme as required in [23].

Below, we formally prove the security of the signature scheme Π. We denote the cost of a signing algorithm Sign' in Π' by $t_{\mathsf{sign}'}$.

Theorem 4. *If Π' is (t', q, ε')-weakly-secure, then the signature Π is (t, q, ε)-fully-secure, where $t \leqslant t' - O(q \cdot t_{\mathsf{sign}'})$ and $\varepsilon \geqslant 2q \cdot \varepsilon'$.*

Proof. See Appendix B. □

In fact, if the signing algorithm Sign' in Π' deterministic, then the *fully-secure* signature scheme Π is strongly unforgeable.

4.3. Parallel composition method

In this section, we show another generic transformation from *weakly-secure* signatures to *fully-secure* signatures.

Before showing the transformation, we define a relation $\mathcal{R} = \{((a, b), c)\}$ that satisfies the following conditions:

- Given a and c (or b and c), b (or a) is determined and can be computed in probabilistic polynomial time (*PPT*);
- Given randomly chosen values a and b, it is hard to find c in *PPT*, such that $((a, b), c) \in \mathcal{R}$.

In fact, this kind of relation can be easily found. Suppose the security parameter is 1^k. For example, given a collision-resistant hash function H: $\{0,1\}^* \to \{0,1\}^k$, $a, b \in \{0,1\}^k$ and $c \in \{0,1\}^*$, we define $((a, b), c) \in \mathcal{R}$, if and only if $a \oplus b = H(c)$.

Obviously, this relation satisfies the definition of $\mathcal{R}$ because: Given $a \in \{0,1\}^k$ and c, $b \in \{0,1\}^k$ is determined and can be computed efficiently; And, randomly choose $a \in \{0,1\}^k$ and $b \in \{0,1\}^k$, it is hard to find c such that $a \oplus b = H(c)$ for the collision-resistant property of the hash function.

In public parameters, relation $\mathcal{R} = \{((a, b), c)\}$ defined above should be given. The generic construction follows:

1. **Gen.** On input security parameter 1^k, invoke $\mathsf{Gen}'(1^k)$ two times and obtain two key pairs (pk_1, sk_1) and (pk_2, sk_2). Output Π's public key $pk = (pk_1, pk_2)$ and secret key $sk = (sk_1, sk_2)$.
2. **Sign.** To sign message m, the signer first chooses m' randomly and computes m'' such that $((m', m''), m) \in \mathcal{R}$. The signer then invokes algorithms $\mathsf{Sign}'(sk_1, m')$ and $\mathsf{Sign}'(sk_2, m'')$. Output $\sigma = (A, B, C)$ as the signature on message m, where $A = \mathsf{Sign}'(sk_1, m')$, $B = \mathsf{Sign}'(sk_2, m'')$, $C = m'$.
3. **Verify**. On input verifying key $pk = (pk_1, pk_2)$, message m, and signature $\sigma = (A, B, C)$, first compute m'' from m and C such that $((C, m''), m) \in \mathcal{R}$ (This can be done from the property of the relation R). Finally, it outputs 1 if and only if $\mathsf{Verify}'(pk_1, C, A) = 1$ and $\mathsf{Verify}'(pk_2, m'', B) = 1$.

It is easy to prove that Π is strongly unforgeable if Π' is deterministic. Below, we formally prove the security of the resulting signature scheme Π, with very tight security reduction to Π'. We also denote the cost of a signing algorithm sign' in Π' by $t_{\mathsf{sign}'}$.

Theorem 5. *The signature scheme Π is (t, q, ε)-fully-secure, provided that Π' is (t', q, ε')-weakly-secure, where $t \leqslant t' - O(q \cdot t_{\mathsf{sign}'})$ and $\varepsilon \geqslant 2\varepsilon'$.*

Proof. See Appendix C. □

4.4. Comparisons

We only compare the efficiency of the last two paradigms, *i.e.*, sequential composition method and parallel composition method, because they are only based on *weakly-secure* signature scheme.

- Key generation phase: The key generation in *fully-secure* signature from the sequential method, is the same with its corresponding key generation of weak signature scheme. And, for the *fully-secure* signature from parallel method, it requires to run the key generation algorithm of weak signature twice. So, the key size is smaller and computation cost is less in sequential method, compared with the parallel method.
- Signing phase: In the first paradigm, the signer should run the key generation algorithm and signing algorithm of weak signature, respectively. In the second paradigm, it requires to run the signing algorithm of weak signature twice. The online computations of both methods in signing phase are the same because it is only required to run signing algorithm of weak signature only once.
- Verification phase: In both paradigms, it requires to run the verification of weak signature scheme twice. So, the computations of verification algorithm are the same.

In conclusion, the sequential method is more suitable for device with small storage such as smart card for its smaller key size. And, the signing algorithm in the sequential composition method requires one key generation of weak signatures. So, if

the computation of this phase is almost the same with signing algorithm of weak signature, then, the sequential method is indeed better than the parallel composition method. Otherwise, from only the computational cost of signing algorithm, the parallel composition method is better. So, we can use different paradigms according to circumstance requirements.

5. Instantiations from sequential composition method with one-time signature scheme

5.1. Fully-secure signature from weak Boneh–Boyen signature

Next, we describe the *fully-secure* signature from the *weakly-secure* signature [5] and $\mathcal{OTS}$. We describe how to get *fully-secure* signature, denoted by S-WBB-OTS, by using the sequential composition method with one-time signature on the weak Boneh–Boyen signature scheme. The public parameters are similar with the weak Boneh–Boyen signature. Let $\mathcal{OTS} = (\mathsf{OGen}, \mathsf{OSign}, \mathsf{OVerify})$ be a one-time signature. Meanwhile, define a collision-resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_p^*$.

1. **Gen:** Pick $x \in \mathbb{Z}_p^*$, compute $y = g^x$. The public key is $pk = (g, y)$ and the secret key is $sk = x$.
2. **Sign:** Given message $m \in \mathbb{Z}_p^*$, the signer first invokes $\mathsf{OGen}(1^k)$ and obtains key pair $(opk, osk) \leftarrow \mathsf{OGen}(1^k)$. Output the signature on m as $\sigma = (A, B, C)$, where $A = g^{\frac{1}{x+H(opk)}},\ B = \mathsf{OSign}(osk, m),\ C = opk$.
3. **Verify:** On input verification key y, message m, and the signature $\sigma = (A, B, C)$, output 1 if and only if $\hat{e}(y \cdot g^{H(opk)}, A) = \hat{e}(g, g)$ and $\mathsf{OVerify}(C, m, B) = 1$. Otherwise, output 0.

Theorem 6. *The* S-WBB-OTS *signature scheme is fully-secure.*

Proof. The result can be derived directly from Theorems 1 and 3. □

Notice that the user's public key consists only one group element y in $\mathbb{G}_1$. So length of the public key is even more shorter than [5]. It requires one point scalar multiplication in $\mathbb{G}_1$ and one-time signature computations in signature generation. In fact, the value A and C can be pre-computed. So the online computation of Sign is only the computation of one-time signature, which is very efficient compared ordinary signature scheme. Verification only requires two pairing computation, one point scalar multiplication in $\mathbb{G}_1$, and an $\mathcal{OTS}$ verification, which is also very efficient. The only disadvantage of this signature scheme is that length of the signature is longer than [5].

5.2. Fully-secure signature from weak GHR signature

In this section, we show the *fully-secure* signature (denoted by S-WGHR-OTS) from *weakly-secure* signature [16] and $\mathcal{OTS} = (\mathsf{OGen}, \mathsf{OSign}, \mathsf{OVerify})$. Define a hash function H which is collision and division intractable satisfies $H : (0,1)^* \to Z_n^*$.

1. **Gen:** Pick two safe primes p and q, compute $n = pq$ as RSA modulus, select $s \in \mathbb{Z}_n^*$. The public key is $pk = (n, s)$ and the secret key is $sk = (p, q)$.
2. **Sign:** To sign a message m, invoke $\mathsf{OGen}(1^k)$ and obtain key pair $(opk, osk) \leftarrow \mathsf{OGen}(1^k)$ of $\mathcal{OTS}$. Output the signature as $\sigma = (A, B, C)$, where $A = s^{\frac{1}{H(opk)}} \bmod n,\ B = \mathsf{OSign}(osk, m),\ C = opk$.
3. **Verify:** On input verification key (n, s), message m, and $\sigma = (A, B, C)$, output 1 if and only if $A^{H(C)} = s \bmod n$ and $\mathsf{OVerify}(C, m, B) = 1$. Otherwise, output 0.

Theorem 7. *The* S-WGHR-OTS *signature scheme is fully-secure.*

Proof. The result can be derived directly from Theorems 2 and 3. □

6. Instantiations from sequential composition method

6.1. Fully-secure signature from weak Boneh–Boyen signature

We describe how to get *fully-secure* signature, denoted by S-WBB, by using the sequential composition method on the weak Boneh–Boyen signature scheme. The public parameters are similar with the weak Boneh–Boyen signature, except a collision resistant hash function $H : \mathbb{G}_1 \to \mathbb{Z}_p^*$ is chosen additionally.

1. **Gen:** Pick $x \in \mathbb{Z}_p^*$, compute $y = g^x$. The public key is y and the secret key is x.
2. **Sign:** Given message $m \in \mathbb{Z}_p^*$, the signer chooses a random $x' \in \mathbb{Z}_p^*$, computes $y' = g^{x'}$, and outputs the signature as $\sigma = (A, B, C)$, where $A = g^{\frac{1}{x+H(y')}}, B = g^{\frac{1}{x'+m}}, C = y'$.
3. **Verify:** On input verification key y, message m, and the signature $\sigma = (A, B, C)$, output 1 if and only if $\hat{e}(y \cdot g^{H(C)}, A) = \hat{e}(g, g)$ and $\hat{e}(y' \cdot g^m, B) = \hat{e}(g, g)$. Otherwise, output 0.

In key generation algorithm, S-WBB scheme needs one exponentiation in group $\mathbb{G}_1$. The signing algorithm costs two exponentiations computations in group $\mathbb{G}_1$ and two inversion computations in $\mathbb{Z}_p^*$. As the value A could be pre-computed, the computations is reduced to only one exponentiation in $\mathbb{G}_1$ and one inversion computation in $\mathbb{Z}_p^*$. In verification algorithm, the value $\hat{e}(g,g)$ can be fixed and published as part of the public key. So, it only needs two pairing and two exponentiations computations.

Compared with the *fully-secure* signature scheme in [5], the key generation algorithm of S-WBB is more efficient. Furthermore, the key size is smaller than [5] because the secret key consists of only one group element. So, it is very suitable for small storage device such as smart card or mobile phone to perform authentication operations. The online computation for signing algorithm in [5] is also one exponentiation in $\mathbb{G}_1$ and one inversion computation in $\mathbb{Z}_p^*$. The computation of online verification in S-WBB requires one more pairing computation compared with [5]. From the above comparison, the S-WBB scheme is very suitable for device with small storage.

Theorem 8. *The* S-WBB *signature scheme is fully-secure.*

Proof. The result can be derived directly from Theorems 1 and 4. □

6.2. Fully-secure signature from weak GHR signature

In this section, we present a *fully-secure* signature, denoted by S-WGHR, from the weak GHR signature scheme [16].

1. **Gen:** On input security parameter 1^k, pick two pairs safe primes (p_1,q_1). Compute $n_1 = p_1q_1$ as a RSA modulus, select $s_1 \in \mathbb{Z}_{n_1}^*$. Meanwhile, choose a division intractability hash function $H_1 : \{0,1\}^* \to \mathbb{Z}_{n_1}^*$. The public key is (n_1,s_1,n_2,s_2,H_1) and the secret key is (p_1,q_1).
2. **Sign:** To sign a message m, choose two pairs safe primes (p_2,q_2), and a random $s_2 \in \mathbb{Z}_{n_2}^*$, compute $n_2 = p_2q_2$. Then, choose a division intractability hash functions and $H_2 : \{0,1\}^* \to \mathbb{Z}_{n_2}^*$ and compute the signature as $\sigma = (A,B,C)$, where $A = s_1^{\frac{1}{H_1(n_2\|s_2\|H_2)}} \bmod n_1$, $B = s_2^{\frac{1}{H_2(m)}} \bmod n_2$, $C = n_2\|s_2\|H_2$.
3. **Verify:** On input verification key (n_1,s_1,H_1), message m, and $\sigma = (A,B,C)$, parse $C = (C_1,C_2,C_3)$. Then, output 1 if and only if $A^{H_1(C)} = s_1 \bmod n_1$ and $B^{C_3(m)} = C_2 \bmod C_1$. Otherwise, output 0.

Theorem 9. *The* S-WGHR *signature scheme is fully-secure.*

Proof. The result can be derived directly from Theorems 2 and 4. □

In key generation algorithm, it requires one multiplications in $\mathbb{Z}_{n_1}^*$. The secret key size is only $[log_2 n_1]$. The signing algorithm needs one exponentiation and inversion computations in $\mathbb{Z}_{n_1}^*$ and $\mathbb{Z}_{n_2}^*$, respectively. As the value A could be pre-computed, the computation is reduced to only one exponentiation and one inversion computation in $\mathbb{Z}_{n_2}^*$. In verification algorithm, it requires one exponentiation computation in $\mathbb{Z}_{n_1}^*$ and $\mathbb{Z}_{n_2}^*$, respectively. Compared to [26], the computations in signing and verification algorithms are almost the same. In key generation algorithm of S-WGHR, the key size is smaller than [26] and it requires less exponentiations to generate key pair.

7. Instantiations from parallel composition method

In the following two instantiations, we will use the concrete relation $\mathcal{R}$ given in Section 4.3: $((a,b),c) \in \mathcal{R}$, if and only if $a \oplus b = H(c)$. The relation should be described in system public parameters, in both following examples.

7.1. Fully-secure signature from weak Boneh–Boyen signature

Denote the following fully-secure signature scheme from the weak Boneh–Boyen by P-WBB. The public parameters are similar with the weak Boneh–Boyen signature, excluding a concrete relation $\mathcal{R}$ given in Section 4.3.

1. **Gen:** Pick $x_1,x_2 \in \mathbb{Z}_p^*$, compute $y_1 = g^{x_1}$ and $y_2 = g^{x_2}$. The public key is (y_1,y_2) and the secret key is (x_1,x_2).
2. **Sign:** Given message $m \in \mathbb{Z}_p^*$, the signer chooses a random $m' \in \mathbb{Z}_p^*$ and computes the signature as $\sigma = (A,B,C)$, where $A = g^{\frac{1}{x_1+m'}}$, $B = g^{\frac{1}{x_2+(H(m)\oplus m')}}$, $C = m'$.
3. **Verify:** On input verification key (y_1,y_2), message m, and the signature $\sigma = (A,B,C)$, output 1 if and only if $\hat{e}(y_1 \cdot g^C, A) = \hat{e}(g,g)$ and $\hat{e}(y_2 \cdot g^{H(m)\oplus C}, B) = \hat{e}(g,g)$. Otherwise, output 0.

In key generation algorithm of P-WBB, it needs two exponentiations in group $\mathbb{G}_1$. The signing algorithm costs two exponentiations computations in group $\mathbb{G}_1$ and two inversion computations in $\mathbb{Z}_p^*$. In verification algorithm, it only needs two pairing and two exponentiations computations as the value $\hat{e}(g,g)$ can be published as part of the public key.

From Theorems 1 and 5, we can get the following result:

Theorem 10. *The* P-WBB *signature scheme is fully-secure.*

The security reduction is the same with Theorem 5.

7.2. Fully-secure signature from weak GHR signature

In this section, we present a *fully-secure* signature, denoted by P-WGHR, from the weak GHR signature [16] with the following advantages: The new scheme does not require the non-standard randomness-finding oracle assumption. The signing algorithm requires less exponentials computation compared to [16].

1. **Gen:** On input security parameter 1^k, pick two pairs safe primes (p_1,q_1), (p_2,q_2). Compute $n_1 = p_1q_1$ and $n_2 = p_2q_2$ as two RSA modulus, select $s_1 \in \mathbb{Z}^*_{n_1}$ and $s_2 \in \mathbb{Z}^*_{n_2}$. Meanwhile, choose two division intractability hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}^*_{n_1}$ and $H_2 : \{0,1\}^* \to \mathbb{Z}^*_{n_2}$. Furthermore, a collision-resistant hash function H: $\{0,1\}^* \to \{0,1\}^k$ is selected. The public key is $(n_1,s_1,n_2,s_2,H_1,H_2,H)$ and the secret key is (p_1,q_1,p_2,q_2).
2. **Sign:** To sign a message m, the signer chooses a random $m' \in \{0,1\}^k$ and computes the signature as $\sigma = (A,B,C)$, where $A = s_1^{\frac{1}{H_1(m')}} \bmod n_1$, $B = s_2^{\frac{1}{H_2(H(m)\oplus m')}} \bmod n_2$, $C = m'$.
3. **Verify:** On input verification key $(n_1,s_1,n_2,s_2,H_1,H_2,H)$, message m, and $\sigma = (A,B,C)$, output 1 if and only if $A^{H_1(C)} = s_1 \bmod n_1$ and $B^{H_2(H(m)\oplus C)} = s_2 \bmod n_2$. Otherwise, output 0.

It requires one multiplication in $\mathbb{Z}^*_{n_1}$ and $\mathbb{Z}^*_{n_2}$ in key generation algorithm, respectively. The signing algorithm needs one exponentiation and inversion computations in $\mathbb{Z}^*_{n_1}$ and $\mathbb{Z}^*_{n_2}$, respectively. The online computation in signing phase could be reduced to only one exponentiation and one inversion computation in $\mathbb{Z}^*_{n_2}$. In verification algorithm, it requires one exponentiation computation in $\mathbb{Z}^*_{n_1}$ and $\mathbb{Z}^*_{n_2}$, respectively.

It is very interesting because this instantiation from the weak GHR signature scheme looks similar to the twin signature scheme in [26]. In fact, the parallel composition paradigm could be viewed as generalization of [26]. First, we define a relation R as follows:

$(a,b),c \in \mathcal{R}$ if and only if $a = c \oplus \mu_1 \| c \oplus \mu_2$, $b = \mu_1 \| \mu_2$ for some μ_1 and μ_2.

It is easy to verify such kind of relation satisfies the definition given in Section 4.3. Based on this given relation and the parallel paradigm, the twin signature scheme [26] could be derived directly from the weak GHR signature scheme.

And, the following result could be derived easily from Theorems 2 and 5. And, security reduction is the same with Theorem 5.

Theorem 11. *The* P-WGHR *signature scheme is fully-secure.*

8. Conclusion

Three new paradigms are proposed to obtain *fully-secure* signature scheme from any scheme satisfies only a weak security notion called existentially unforgeable against generic chosen message attacks. The sequential composition (with one-time signature) methods are very efficient in key generation algorithm compared to the parallel composition method. However, if the computation cost in the key generation algorithm of weak signature needs more than the weak signature's signing algorithm, then, the signing algorithm is more efficient in the parallel paradigm. Therefore, different paradigm can be adopted in applications according to different requirements.

We also present several instantiations which are converted from two previous *weakly-secure* signature schemes and are *fully-secure* without random oracles. The comparisons with the previous secure signatures show that our paradigms are very efficient.

Acknowledgements

This work is partially supported by the National Natural Science Foundation of China (No. 61100224) and Foundation for Distinguished Young Talents in Higher Education of Guangdong, China. The second author is supported by the National Natural Science Foundation of China (No. 61070168). The third author is supported by the National Natural Science Foundation of China (No. 60970144).

Appendix A. Proof of Theorem 3

Proof. Given any adversary $\mathcal{A}$ attacking Π in an adaptive chosen message attack, we construct an adversary $\mathcal{A}'$ breaking Π' in a weak chosen message attack or breaking $\mathcal{OTS}$. After given public key pk of Π, $\mathcal{A}$ queries the signing oracle of Π on messages m_i adaptively and gets q_S signatures $\sigma_i = (A_i,B_i,C_i)$ for $1 \leqslant i \leqslant q_S$. After the signature queries, $\mathcal{A}$ outputs a forged signature on a new m^* as $\sigma^* = (A^*,B^*,C^*)$.

There are two types of forgeries. The reduction works differently for each forger type. Therefore, initially $\mathcal{A}'$ will choose a random bit $b_{code} \in \{1,2\}$ that indicates its guess for the type of forger that $\mathcal{A}$ will emulate. The simulation proceeds differently for each b_{code}.

Type 1 forgery. $C^* \neq C_i$ for $1 \leqslant i \leqslant q_S$.

Algorithm $\mathcal{A}'$ first picks a random bit b_{code}. If $b_{code} = 1$, we construct an algorithm $\mathcal{A}'$ to break Π'. $\mathcal{A}'$ first invokes $\mathsf{OGen}(1^k)$ and gets q_S key pairs $(opk_i, osk_i) \leftarrow \mathsf{OGen}(1^k)$ for $\mathcal{OTS}$ (Assume $\mathcal{A}$ makes at most q_S queries to signing oracle), and sends the q_S values opk_i, for $1 \leqslant i \leqslant q_S$, to challenger for signature queries of Π' before the parameters publication of Π'. Then $\mathcal{A}'$ gets public key pk of Π' and q_S signatures σ'_i on the q_S messages opk_i for $1 \leqslant i \leqslant q_S$.

Then $\mathcal{A}'$ sends the public key pk to the adversary $\mathcal{A}$ as the public key of Π. $\mathcal{A}$ then queries the signing oracle of Π on messages m_i adaptively for $1 \leqslant i \leqslant q_S$. $\mathcal{A}'$ answers the signature query as follows: $\sigma_i = (A_i, B_i, C_i)$, where $A_i = \sigma'_i$ from the challenger, $B_i = \mathsf{OSign}(osk_i, m_i)$, $C_i = opk_i$.

After the signature queries, $\mathcal{A}$ outputs a forged signature on a new message m^* as $\sigma^* = (A^*, B^*, C^*)$. Because $C^* \neq C_i$ for $1 \leqslant i \leqslant q_S$, then $\mathcal{A}'$ can output a forged Π' signature as $\sigma = A^*$ on a new message C^* and break the signature scheme Π'.

Type 2 forgery: $C^* = C_i$ for some i, $1 \leqslant i \leqslant q_S$.

If $b_{code} = 2$, we construct an algorithm $\mathcal{A}'$ to break $\mathcal{OTS}$. $\mathcal{A}'$ is given opk^* from challenger as the challenge public key for $\mathcal{OTS}$. Then it randomly generates $(pk, sk) \leftarrow \mathsf{Gen}'(1^k)$ of Π'. $\mathcal{A}'$ then gets the key pair (pk, sk) of Π and sends the public key pk to $\mathcal{A}$. $\mathcal{A}'$ also chooses a random $\kappa \in [1, q_S]$ and keeps it secret.

Then $\mathcal{A}$ queries the signing oracle of Π on messages m_i adaptively for $1 \leqslant i \leqslant q_S$. $\mathcal{A}'$ answers the signature query as follows: if $i \neq \kappa$, $\mathcal{A}'$ computes one-time key pair (opk_i, osk_i), returns the signature as $\sigma_i = (A_i, B_i, C_i)$, where $A_i = \mathsf{Sign}'(sk, opk_i)$, $B_i = \mathsf{OSign}(osk_i, m_i)$, $C_i = opk_i$. Otherwise, if $i = \kappa$, $\mathcal{A}'$ sends m_i to the challenger for one-time signature with respect to public key opk^* and gets the one-time signature B_i on message m_i. Then $\mathcal{A}'$ answer the signature query as $\sigma_i = (A_i, B_i, C_i)$, where $B_i = \mathsf{Sign}'(sk, opk^*)$ and $C_i = opk^*$.

After the signature queries, $\mathcal{A}$ outputs a forged signature on a new message m^* as $\sigma^* = (A^*, B^*, C^*)$, where $C^* = C_i$ for some $1 \leqslant i \leqslant q_S$. If $i \neq \kappa$, $\mathcal{A}'$ aborts and fails. If $i = \kappa$ (with success probability $\frac{1}{q_S}$), then $C^* = opk^*$. Meanwhile, $B^* \neq B_i$ because $m^* \neq m_i$. This implies that $\mathcal{A}'$ can output a forged one-time signature B^* on a new message m^* with respect to opk^* and break the one-time signature scheme $\mathcal{OTS}$. □

Appendix B. Proof of Theorem 4

Proof. Given any adversary $\mathcal{A}$ attacking Π in an adaptive chosen message attack, we construct an adversary $\mathcal{A}'$ breaking Π' in weak chosen message attacks. After given public key pk of Π, $\mathcal{A}$ queries the signing oracle of Π on messages m_i adaptively and gets q signatures $\sigma_i = (A_i, B_i, C_i)$ for $1 \leqslant i \leqslant q$. After the signature queries, $\mathcal{A}$ outputs a forged signature on a new m^* as $\sigma^* = (A^*, B^*, C^*)$. There are two types of forgeries.

Type 1 forgery: $C^* \neq C_i$ for $1 \leqslant i \leqslant q$.
Type 2 forgery: $C^* = C_i$ for some i, $1 \leqslant i \leqslant q$.

The reduction works differently for each forger type. Therefore, initially $\mathcal{A}'$ will choose a random bit $b_{code} \in \{1,2\}$ that indicates its guess for the type of forger. The simulation proceeds differently for each b_{code}. If $b_{code} = 1$, we construct an algorithm $\mathcal{A}'$ to break Π' as follows:

Simulation of key generation. $\mathcal{A}'$ first invokes $\mathsf{Gen}'(1^k)$ and gets q key pairs $(pk_i, sk_i) \leftarrow \mathsf{Gen}'(1^k)$ (Assume that $\mathcal{A}$ makes at most q queries to signing oracle), and sends the q values pk_i, for $1 \leqslant i \leqslant q$, to challenger for signature queries of Π' before the parameters publication of Π'. Then $\mathcal{A}'$ gets public key pk of Π' and q signatures $\sigma'_i = \mathsf{Sign}'(sk, pk_i)$ on the q messages pk_i, with respect to pk, for $1 \leqslant i \leqslant q$. $\mathcal{A}'$ sends the public key pk to the adversary $\mathcal{A}$ as the public key of Π.

Simulation of signing oracle. $\mathcal{A}$ then queries the signing oracle of Π on messages m_i adaptively for $1 \leqslant i \leqslant q$. $\mathcal{A}'$ answers the signature query as $\sigma_i = (A_i, B_i, C_i)$, where $A_i = \sigma'_i$ from the challenger, $B_i = \mathsf{Sign}'(sk_i, m_i)$, $C_i = pk_i$.

Forgery. After the signature queries, $\mathcal{A}$ outputs a forged signature on a new message m^* as $\sigma^* = (A^*, B^*, C^*)$. If $C^* \neq C_i$ for $1 \leqslant i \leqslant q$, then $\mathcal{A}'$ can output a forged Π' signature as $\sigma = A^*$ on a new message C^* and break the signature scheme Π'. Otherwise, $\mathcal{A}'$ aborts.

If $b_{code} = 2$, we construct an algorithm $\mathcal{A}'$ to break Π' as follows:

Simulation of key generation. $\mathcal{A}'$ randomly generates $(pk, sk) \leftarrow \mathsf{Gen}'(1^k)$ of Π'. $\mathcal{A}'$ then sets $\Pi's$ key pair as (pk, sk) and sends the public key pk to $\mathcal{A}$. $\mathcal{A}'$ also chooses a random $\kappa \in [1, q]$ and keeps it secret.

Simulation of signing oracle. $\mathcal{A}$ queries the signing oracle of Π on messages m_i adaptively for $1 \leqslant i \leqslant q$. $\mathcal{A}'$ answers the signature query as follows: if $i \neq \kappa$, $\mathcal{A}'$ first invokes $\mathsf{Gen}'(1^k)$ and gets key pair $(pk_i, sk_i) \leftarrow \mathsf{Gen}'(1^k)$. Then, it returns the simulated signature on messages m_i as $\sigma_i = (A_i, B_i, C_i)$, where $A_i = \mathsf{Sign}'(sk, pk_i)$, $B_i = \mathsf{Sign}'(sk_i, m_i)$, $C_i = pk_i$. Otherwise, if $i = \kappa$, $\mathcal{A}'$ sends m_i to the challenger for signature of Π', and gets the challenge public key pk^* of Π' and signature $\mathsf{Sign}'(sk^*, m_i)$ of m_i with respect to pk^*. Then $\mathcal{A}'$ answers the signature query as $\sigma_i = (A_i, B_i, C_i)$, where $A_i = \mathsf{Sign}'(sk, pk^*)$, $B_i = \mathsf{Sign}'(sk^*, m_i)$ and $C_i = pk^*$.

Forgery. After the signature queries, $\mathcal{A}$ outputs a forged signature on a new message m^* as $\sigma^* = (A^*, B^*, C^*)$, where $C^* = C_i$ for some $1 \leqslant i \leqslant q$.

If $i \neq \kappa$, $\mathcal{A}'$ aborts and fails. Otherwise, if $i = \kappa$, then $c^* = pk^*$. This implies that $\mathcal{A}'$ can output a forged signature B^* on a new message m^* with respect to pk^* and break the signature scheme Π'.

We define two events, E_1 and E_2, which denotes type 1 forgery and type 2 forgery occurs, respectively. As $prob[E_1] + prob[E_2] = prob[\mathcal{A}\text{ wins}]$. Since $\mathcal{A}$ wins with probability ε, it follows that one of the two events occurs with probability at least $\varepsilon/2$. It is easy to see that the success probability of $\mathcal{A}'$ under the conditions that event E_1 occurs is $\frac{1}{2} \cdot \text{prob}[E_1]$. In the type 2 forgery simulation, success guess of γ is $\frac{1}{q}$. So the success probability of $\mathcal{A}'$ under the conditions that event E_2 occurs is $\frac{1}{2q}\ prob[E_2]$. Therefore, if $\mathcal{A}$ wins with probability ε, the signature scheme Π' with probability at least $\frac{\varepsilon}{2q}$. □

Appendix C. Proof of Theorem 5

Proof. Given any adversary $\mathcal{A}$ attacking Π in an adaptive chosen message attack, we construct an adversary $\mathcal{A}'$ breaking Π' in weak chosen message attacks. After given public key pk of Π, $\mathcal{A}$ queries the signing oracle of Π on messages m_i adaptively and gets q signatures $\sigma_i = (A_i, B_i, C_i)$ for $1 \leqslant i \leqslant q$. After the signature queries, $\mathcal{A}$ outputs a forged signature on a new m^* as $\sigma^* = (A^*, B^*, C^*)$.

There are also two types of forgeries:

Type 1 forgery: $C^* \neq C_i$ for $1 \leqslant i \leqslant q$.
Type 2 forgery: $C^* = C_i$ for some i, $1 \leqslant i \leqslant q$.

The reduction works differently for each forger type. Therefore, initially $\mathcal{A}'$ will choose a random bit $b_{code} \in \{1,2\}$ that indicates its guess for the type of forger that $\mathcal{A}$ will emulate. The simulation proceeds differently for each b_{code}.

If $b_{code} = 1$, we construct an algorithm $\mathcal{A}'$ to break Π' as follows:

Simulation of key generation. $\mathcal{A}'$ first invokes $\mathsf{Gen}'(1^k)$ and gets key pair $(pk_2, sk_2) \leftarrow \mathsf{Gen}'(1^k)$. Then $\mathcal{A}'$ chooses q random values $m'_1, \ldots, m'_q$ (Assume $\mathcal{A}$ makes at most q queries to signing oracle), and sends the q values m'_i, for $1 \leqslant i \leqslant q$, to challenger for signature queries of Π' before the parameters publication of Π'. Then $\mathcal{A}'$ gets its challenge public key $\overline{pk}$ of Π' and q signatures $\sigma'_i = \mathsf{Sign}'(\overline{sk}, m'_i)$ on the q messages m'_i, with respect to $\overline{pk}$, for $1 \leqslant i \leqslant q$. Then $\mathcal{A}'$ sets the public key of Π as $pk = (pk_1, pk_2)$, where $pk_1 = \overline{pk}$, and sends the public key pk to the adversary $\mathcal{A}$.

Simulation of signing oracle. $\mathcal{A}$ then queries the signing oracle of Π on messages m_i adaptively for $1 \leqslant i \leqslant q$. $\mathcal{A}'$ answers the signature query as follows:

- From the first property of the given relation $\mathcal{R}$, $\mathcal{A}'$ could compute m''_i such that $((m'_i, m''_i), m_i) \in \mathcal{R}$;
- Then, it computes $B_i = \mathsf{Sign}'(sk_2, m''_i)$;
- Finally, outputs the signature $\sigma_i = (A_i, B_i, C_i)$, where $A_i = \sigma'_i$ from challenger, and $C_i = m'_i$.

Forgery. After the signature queries, $\mathcal{A}$ outputs a forged signature on a new message m^* as $\sigma^* = (A^*, B^*, C^*)$. Because $m^* \neq m_i$ for $1 \leqslant i \leqslant q$, then if $C^* = m'_i$ for some i, $\mathcal{A}'$ aborts and fails. Otherwise, $\mathcal{A}'$ can output a forged Π' signature as $\sigma = A^*$ of a new message C^* and break the signature scheme Π', with the challenge public key $\overline{pk}$.

If $b_{code} = 2$, we construct an algorithm $\mathcal{A}'$ to break Π' in another way:

Simulation of key generation. $\mathcal{A}'$ randomly generates $(pk_1, sk_1) \leftarrow \mathsf{Gen}'(1^k)$ of Π'. Then $\mathcal{A}'$ chooses q random values $m''_1, \ldots, m''_q$, and sends the q values m''_i, for $1 \leqslant i \leqslant q$, to challenger for signature queries of Π' before the parameters publication of Π'. Then $\mathcal{A}'$ gets its challenge public key $\overline{pk}$ of Π' and q signatures $\sigma'_i = \mathsf{Sign}'\left(\overline{sk}, m''_i\right)$ of the q messages m''_i, with respect to $\overline{pk}$, for $1 \leqslant i \leqslant q$. Then $\mathcal{A}'$ sets the public key of Π as $pk = (pk_1, pk_2)$, where $pk_2 = \overline{pk}$, and sends the public key pk to the adversary $\mathcal{A}$.

Simulation of signing oracle. $\mathcal{A}$ then queries the signing oracle of Π on messages m_i adaptively for $1 \leqslant i \leqslant q$. $\mathcal{A}'$ answers the signature query as follows: First, from the first property of relation $\mathcal{R}$, $\mathcal{A}'$ computes m'_i such that $((m'_i, m''_i), m_i) \in \mathcal{R}$. Then, $\mathcal{A}'$ outputs simulated signature on message m_i as $\sigma_i = (A_i, B_i, C_i)$, where $A_i = \mathsf{Sign}'(sk_1, m'_i)$, $B_i = \sigma'_i$, $C_i = m'_i$.

Forgery. After the signature queries, $\mathcal{A}$ outputs a forged signature on a new message m^* as $\sigma^* = (A^*, B^*, C^*)$. By using the first property of relation $\mathcal{R}$ again, $\mathcal{A}'$ could compute $m^{*''}$ from m^* and C^*, such that $((C^*, m^{*''}), m^*) \in \mathcal{R}$.

Recall that in this kind of forgery, $C^* = C_i$ for some i. Because $m^* \neq m_i$ for $1 \leqslant i \leqslant q$, and m'_i, m''_i are chosen randomly by the simulator, we have $m^{*''} \neq m''_i$ from the second property of the defined relation $\mathcal{R}$. This proof, in fact, shows that the signature scheme prevents the attack from the adversary that just combine the first part in one signature for message M and the second part in the other signature for message M'.

So, $\mathcal{A}'$ can output a forged Π' signature as $\sigma = A^*$ on a new message $m^{*''}$ and break the signature scheme Π', with respect to the challenge public key $\overline{pk}$. □

References

[1] M. Bellare, S. Micali, How to Sign Given Any Trapdoor Function, 39, ACM, 1992. pp. 214–233.
[2] M. Bellare, P. Rogaway, Random Oracles are Practical: A Paradigm for Designing Efficient Protocols, CCS, ACM Press, 1993. pp. 62–73.

[3] M. Bellare, P. Rogaway, The exact security of digital signatures-how to sign with RSA and Rabin, in: U.M. Maurer (Ed.), EUROCRYPT, LNCS, vol. 1070, Springer, 1996, pp. 399–416.
[4] M. Bellare, S. Shoup, Two-tier signatures, strongly unforgeable signatures, and Fiat–Shamir without random oracles, in: T. Okamoto, X. Wang (Eds.), PKC, LNCS, vol. 4450, Springer, 2007, pp. 201–216.
[5] D. Boneh, X. Boyen, Short signatures without random oracles, in: C. Cachin, J.L. Camenisch (Eds.), EUROCRYPT, LNCS, vol. 3027, Springer, 2004, pp. 56–73.
[6] D. Boneh, B. Lynn, H. Shacham, Short signatures from the Weil pairing, in: C. Boyd (Ed.), ASIACRYPT, LNCS, vol. 2248, Springer, 2001, pp. 514–532.
[7] D. Boneh, E. Shen, B. Waters, Strongly unforgeable signatures based on computational Diffie–Hellman, in: M. Yung, Y. Dodis, A. Kiayias, T.G. Malkin (Eds.), PKC, LNCS, vol. 3958, Springer, 2006, pp. 229–240.
[8] J. Camenisch, A. Lysyanskaya, Signature schemes and anonymous credentials from bilinear maps, in: M.K. Franklin (Ed.), CRYPTO, LNCS, vol. 3152, Springer, 2004, pp. 56–72.
[9] R. Canetti, O. Goldreich, S. Halevi, The Random Oracle Methodology, Revisited, STOC 1998, ACM, 1998, pp. 207–221.
[10] J.-S. Coron, D. Naccache, Security analysis of the Gennaro–Halevi–Rabin signature scheme, in: B. Preneel (Ed.), EUROCRYPT, LNCS, vol. 1807, Springer, 2000, pp. 91–101.
[11] R. Cramer, I. Damgård, Secure signature schemes based on interactive protocols, in: D. Coppersmith (Ed.), CRYPTO, LNCS, vol. 963, Springer, 1995, pp. 297–310.
[12] R. Cramer, V. Shoup, Signature schemes based on the strong RSA assumption, ACM TISSEC 3 (3) (2000) 161–185.
[13] Q. Dong, X. Li, Y. Liu, Two extensions of the ring signature scheme of Rivest–Shamir–Taumann, Information Sciences 188 (2012) 338–345.
[14] S. Even, O. Goldreich, S. Micali, On-line/off-line digital signatures, Journal of Cryptology 9 (1996) 35–67.
[15] A. Fiat, A. Shamir, How to prove yourself: practical solutions to identification and signature problems, in: A.M. Odlyzko (Ed.), CRYPTO, LNCS, vol. 263, Springer, 1986, pp. 186–194.
[16] R. Gennaro, S. Halevi, T. Rabin, Secure Hash-and-Sign signatures without the random oracle, in: J. Stern (Ed.), EUROCRYPT, LNCS, vol. 1592, Springer, 1999, pp. 123–139.
[17] E.-J. Goh, S. Jarecki, A signature scheme as secure as the Diffie–Hellman problem, in: E. Biham (Ed.), EUROCRYPT, LNCS, vol. 2656, Springer, 2003, pp. 401–415.
[18] S. Goldwasser, S. Micali, R.L. Rivest, A digital signature scheme secure against adaptive chosen-message attacks, SIAM Journal of Computing 17 (2) (1988) 281–308.
[19] S. Goldwasser, R. Ostrovsky, Invariant signatures and non-interactive zero-knowledge proofs are equivalent, in: E.F. Brickell (Ed.), CRYPTO, LNCS, vol. 740, Springer, 1992, pp. 228–239.
[20] D. Hofheinz, T. Jager, E. Kiltz, Short signatures from weaker assumptions, in: ASIACRYPT 2011, LNCS 7073, 2011, pp. 647–666.
[21] Q. Huang, D.S. Wong, J. Li, Y. Zhao, Generic transformation to strongly unforgeable signatures, Journal of Computer Science and Technology 23 (2) (2008) 1–17. Extended abstract in ACNS 2007, Springer, LNCS 4521, pp. 1–17.
[22] L. Lamport, Constructing Digital Signatures from a One Way Function, Technical Report CSL-98, SRI International, 1979.
[23] J. Li, Y.Y. Chan, Y. Wang, A generic construction of secure signatures without random oracles, in: M.L. Gavrilova, O. Gervasi, et al. (Eds.), ICCSA, LNCS, vol. 3982, Springer, 2006, pp. 309–317.
[24] J. Li, K. Kim, F. Zhang, D. Wong, Generic security amplifying methods of ordinary digital signatures, in: S. Bellovin, R. Gennaro, et al. (Eds.), ACNS, LNCS, vol. 5037, Springer, 2008, pp. 224–241.
[25] H. Krawczyk, T. Rabin, Chameleon hashing and signatures, in: NDSS 2000, Internet Society, eprint.iacr.org/1998/010.
[26] D. Naccache, D. Pointcheval, J. Stern, Twin signatures: an alternative to the hash-and-sign paradigm, in: ACM Conference on Computer and Communications Security 2001, 2001, pp. 20–27.
[27] M. Naor, M. Yung, Universal one-way Hash functions and their cryptographic applications, ACM Symposium on Theory of Computing, ACM Press, 1989.
[28] D. Pointcheval, J. Stern, Security arguments for digital signatures and blind signatures, Journal of Cryptology 13 (3) (2000) 361–396.
[29] R. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signature and public key cryptosystems, Communication of ACM (1978) 120–126.
[30] C.P. Schnorr, Efficient signature generation by smart cards, Journal of Cryptology 4 (1991) 161–174.
[31] S. Seo, K.Y. Choi, J.Y. Hwang, S. Kim, Efficient certificateless proxy signature scheme with provable security, Information Sciences 188 (2012) 322–337.
[32] A. Shamir, Y. Tauman, Improved online/offline signature schemes, in: CRYPTO, LNCS, vol. 2139, Springer, 2001, pp. 355–367.
[33] R. Steinfeld, J. Pieprzyk, H. Wang, How to strengthen any weakly unforgeable signature into a strongly unforgeable signature, in: M. Abe (Ed.), CT-RSA, LNCS, vol. 4377, Springer, 2007, pp. 357–371.
[34] B. Waters, Efficient identity-based encryption without random oracles, in: R. Cramer (Ed.), EUROCRYPT, LNCS, vol. 3494, Springer, 2005, pp. 114–127.
[35] J. Yu, R. Hao, F. Kong, X. Cheng, J. Fan, Y. Chen, Forward-secure identity-based signature: security notions and construction, Information Sciences 181 (3) (2011) 648–660.
[36] F. Zhang, R. Safavi-Naini, W. Susilo, An efficient signature scheme from bilinear pairings and its applications, in: F. Bao, R. Deng, J. Zhou (Eds.), PKC, LNCS, vol. 2947, Springer, 2004, pp. 277–290.