Generic Security-Amplifying Methods of Ordinary Digital Signatures

Jin Li[1], Kwangjo Kim[1], Fangguo Zhang[2], and Duncan S. Wong[3]

[1] International Research center for Information Security (IRIS)
Information and Communications University(ICU)
103-6 Munji-Dong, Yuseong-Gu, Daejeon, 305-732, Korea
{jjl,kkj}@icu.ac.kr

[2] School of Information Science and Technology
Sun Yat-Sen University, Guangzhou, 510275, P.R.China
isdzhfg@mail.sysu.edu.cn

[3] Department of Computer Science
City University of Hong Kong, Hong Kong, China
duncan@cs.cityu.edu.hk

Abstract. We describe two new paradigms on how to obtain ordinary signatures that are secure against existential forgery under adaptively chosen message attacks (*fully-secure*, in short), from any signatures satisfy only a weak security notion called existentially unforgeable against weak chosen message attacks (*weakly-secure*, in short). The new transformations from a *weakly-secure* signature scheme to *fully-secure* signature scheme are generic, simple, and provably secure in the standard model. Moreover, these two new paradigms are built only on *weakly-secure* signatures. They are different from the previous methods, which also relied on some other cryptographic protocols or non-standard models.

By using two new paradigms, several efficient instantiations without random oracles are also presented, which are based on two previous *weakly-secure* signature schemes. These *fully-secure* signature schemes have many special interesting properties compared with the previous related signature schemes.

Keywords: Signature, Weak Chosen Message Attack, q-SDH Assumption, Strong-RSA Assumption, Strong Unforgeability.

1 Introduction

Digital signature is a central cryptographic primitive. The standard definition on the security of signature scheme was given by Goldwasser, Micali, and Rivest [18]. In fact, in terms of the goals and resources of the adversary, there are many security models can be formed. Compared to the standard security model, there are also many weak security models. Signatures in these weak security models are not sufficient in many practical applications. Among these weak security models, we will focus on the weak security model mentioned in [5,18], which is called existentially unforgeable against generic chosen message attack (or, weak chosen message attack). The signatures that satisfy this security model are called *weakly-secure* signatures in this paper. In this security model, it requires the adversary to submit all signature queries before the signer's public key is published. More detailed definition will be given in Section 2. Obviously, because of the limitation of signature queries, it is insecure in many practical applications. However, in our paper, we will show its applications in the constructions of standard signatures and strongly secure signatures.

Since the standard definition on the security of signature schemes was given [18], there are many attempts to design practical and provably secure signature schemes in this security model. 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 [6,17,33], factoring [3]. Some constructions are based on other assumptions [29,34]. These schemes are very efficient, but their security can be proven only in the random oracle model. However, Canetti *et al.* [9] proved that some popular cryptosystems previously proved secure in the random oracle are actually provably insecure when the random oracle is instantiated by any real-world hashing functions. Over the years, several signature schemes were proposed in the standard model based on some stronger complexity assumptions such as [5,8,13,16]. Among them, the most efficient schemes are based on the Strong-RSA assumption [13,16] and q-strong Diffie-Hellman (q-SDH) assumption [5]. These assumptions are cryptographically stronger than the computational Diffie-Hellman, factoring, and RSA assumptions. The reason is that in order to simulate the signing oracle for the adversary in the proof, the simulator has to get some additional auxiliary inputs.

There are also many generic constructions of signatures. Most of them are based on the basic cryptographic primitive, such as (trapdoor) one-way functions [1,21]. There are also many generic constructions from other cryptographic protocols such as non-interactive zero-knowledge [19] and [12,15] *et al.* Among them, the most famous is the Fiat-Shamir (FS) transform [15]. Recall that the FS-transform [15] is a way to obtain a signature scheme from a three-move identification Σ protocol (honest-verifier zero-knowledge protocol) by collapsing the interaction via a hash function. But its security relies on the random oracle model. In order to avoid the usage of random oracle model, from the Σ protocol, Cramer *et al.* [12] gave another generic transform. The transform also uses hash function which does not act as a random oracle in the proving process. 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.

Compared to the standard security notion of existential unforgeability, there is another strong security notion which is called strongly existential unforgeability. Recently, this notion was concerned by many papers, such as [7,20,32].

1.1 Our Results

Firstly, we present two new paradigms to transform any *weakly-secure* signature schemes into *fully-secure* signature schemes. More precisely, the two paradigms are called sequential composition and parallel composition method, respectively. The new transformations from a *weakly-secure* signature scheme to *fully-secure* one are generic, simple and provably in the standard model. The goal is to obtain constructions that are based on standard assumptions and are efficient. They have interest from both theoretical and practical perspective.

– **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 and used to sign the other public key, which is generated in signing algorithm.
– **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 paradigm, *i.e.*, the sequential composition method, is very efficient in key generation algorithm compared to the second. However, the signing algorithm is more efficient in the second paradigm. So, we can use different paradigm in different circumstances according to its requirements. This is a coincidence that, when instantialized from weak signature scheme [16], the construction will be similar to twin signature scheme [25]. In fact, our second paradigm can be viewed as generalization and extension of the twin signature scheme [25]. And, in both paradigms, if the signing algorithm in the weak signature is deterministic, the resulted *fully-secure* signature is strongly unforgeable secure.

1.2 Organization

In the next section, the definitions of variant signatures are given. Then, two previous instantiations of *weakly-secure* signature schemes are reviewed in Section 3. In Section 4, we propose our two generic transformations techniques. The security proof for these two transformations are given in Appendix. In Section 5, two instantiations from sequential composition method are presented based on two previous *weakly-secure* signature schemes. In Section 6, we present the two instantiations from parallel composition method. We discuss the efficiency of our two generic transformation methods and instantiations by comparing them with the previous signatures. Finally, the conclusion will be made.

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 the goals of the adversary, it can be divided into four categories [18]:

– *Total break*: This is the most serious attack, in which the adversary is able to disclose the secret key of the signer.
– *Universal forgery*: The adversary is able to sign any given messages.
– *Existential forgery*: The adversary is able to provide a signature on a new message whose signature has not been seen.
– *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 [18]. We focus ourselves on two kinds of message attacks:

– *Weakly chosen message attack*: The adversary is allowed to obtain signatures from the signer for a chosen list of messages before it attempts to break the scheme. These messages chosen by the adversary must be given to the signer before seeing the signer's public key.
– *Adaptively chosen message attack*: The adversary is allowed to request signatures of messages chosen by itself. These messages may not only depend on signer's public key, but also depend on the previous obtained signatures.

2.1 Unforgeability

By combining the different goals of the adversary and various resource available to the adversary, many security notions for signature schemes can be derived. The standard notion of security for a signature scheme is called existential unforgeability under adaptively chosen message attacks (*fully-secure* signatures) [18], which is defined through the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:

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, $1 \le i \le q$, $\mathcal{A}$ 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.*, $m^* \notin \{m_1, m_2, \cdots, m_q\}$.

Definition 1 (Unforgeability). *A signature scheme Π =(Gen, Sign, 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.*

If we lower down the adversary's goal to strong existential forgeability and keep its ability unchanged, we can get a stronger definition compared to existential unforgeability against adaptive chosen message attacks:

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.

If we lower the adversary's ability to weak chosen message attack while keeping the goal of the adversary unchanged compared to the standard security notion, we can get a weaker definition compared to existential unforgeability against adaptive chosen message attacks:

2.3 Weak Unforgeability

The difference between this security notion with the standard security [18] is that here it requires that the adversary should submit all messages for signature queries before the public key is seen. 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^*.

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 $1 \le i \le q$.
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.*, $m^* \notin \{m_1, m_2, \cdots, m_q\}$.

Definition 2 (Weak Unforgeability). *A signature scheme Π = (Gen, Sign, 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 $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ 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 [6,34], 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]. Let $(\mathbb{G}_1, \mathbb{G}_2)$ be bilinear groups where the order of $\mathbb{G}_1$ and $\mathbb{G}_2$ is p. As usual, g is a generator of $\mathbb{G}_1$.

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 outputs the signature on m as $\sigma = g^{\frac{1}{x+m}}$.
3. **Verify:** On input verification key y, message m, and the signature σ, output 1 if and only if $\hat{e}(y \cdot g^m, \sigma) = \hat{e}(g, g)$. Otherwise, output 0.

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, Halevi and Rabin 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 $a_1, a_2, \cdots, a_k$ 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. Then, the scheme in [16] can only be proven *weakly-secure* without the randomness-finding oracle. This problem was also addressed in [11], which presented an extension to [16] without relying on this non-standard assumption.

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 safe primes p and q, compute $n = pq$ as RSA modulus, a hash function H, and select $s \in \mathbb{Z}_n^*$. 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, output 1 if and only if $\sigma^{H(m)} = s \bmod n$. 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] □

As stated in [11,16], division-intractable hash functions can be constructed from collision-intractable hash functions [26].

4 Fully-Secure Signatures from Weakly-Secure Signatures

4.1 Related Work

There are two main techniques in order to get *fully-secure* signatures from *weakly-secure* signatures:

– 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 [5,34].
– 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 [5,14,24,31].

Compared to *fully-secure* signatures, the construction of *weakly-secure* signatures is relatively easy (Obvious, every *fully-secure* signature scheme also satisfies security notion of *weakly-secure* signature). There have several *weakly-secure* signature schemes in the open literature.

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 σ=(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 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 [4,22]. However, only *weakly-secure* signatures are required here, instead of fully secure signature scheme or one-time signature scheme as required in [4,22]. This could be viewed as improvements to the results [4,22].

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 3. *If Π' is (t', q, ε')-weakly-secure, then the signature Π is (t, q, ε)-fully-secure, where $t \leq t' - O(q \cdot t_{\mathsf{sign}'})$ and $\varepsilon \geq 2q \cdot \varepsilon'$.* [1]

Proof. See Appendix A. □

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);

[1] In fact, the key pair $(pk', sk') \leftarrow \Pi'$ generated in the signing algorithm is only used to sign message for one-time. So, in fact, (pk', sk') can be generated from weakly-secure signatures satisfies only $(t', 1, \varepsilon')$-weakly-secure, which is easier to be constructed compared to (t', q, ε')-weakly-secure signatures.

– 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 σ=(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 4. *The signature scheme Π is (t, q, ε)-fully-secure, provided that Π' is (t', q, ε')-weakly-secure, where $t \leq t' - O(q \cdot t_{\mathsf{sign}'})$ and $\varepsilon \geq 2\varepsilon'$.*

Proof. See Appendix B. □

4.4 Comparison of Two Paradigms

– 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 requires 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.

After presenting two paradigms, we will describe several instantiations converted from the *weakly-secure signature* schemes [5,16].

5 Instantiations from Sequential Composition Method

5.1 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 $H : \mathbb{G}_1 \rightarrow \mathbb{Z}_p^*$ 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 σ=(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 5. *The S-WBB signature scheme is fully-secure.*

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

5.2 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 $n_1 = p_1 q_1$ as a RSA modulus, select $s_1 \in \mathbb{Z}_{n_1}^*$. Meanwhile, choose a division intractability hash function $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_{n_1}^*$. 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 $n_2 = p_2 q_2$. Then, choose a division intractability hash functions and $H_2 : \{0,1\}^* \rightarrow \mathbb{Z}_{n_2}^*$ and compute the signature as $\sigma{=}(A, B, C)$, where $A = s_1^{\frac{1}{H_1(n_2 \| s_2 \| H_2)}} \bmod n_1$, $B = s_2^{\frac{1}{H_2(m)}} \bmod n_2$, $C = n_2 \parallel s_2 \parallel H_2$.
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 6. *The S-WGHR signature scheme is fully-secure.*

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

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 [25], the computations in signing and verification algorithms are almost the same. In key generation algorithm of S-WGHR, the key size is smaller than [25] and it requires less exponentiations to generate key pair.

6 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.

6.1 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 σ=$(A,\ B,\ C)$, 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 4, we can get the following result:

Theorem 7. *The P-WBB signature scheme is fully-secure.*

The security reduction is the same with Theorem 4.

6.2 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 [16]. 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 σ=(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 σ=(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 [25]. In fact, the parallel composition paradigm could be viewed as generalization of [25]. First, we define a relation R as follows:

$(a, b), c) \in \mathcal{R}$ if and only if $a = c \oplus \mu_1 \parallel c \oplus \mu_2$, $b = \mu_1 \parallel \mu_2$ 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 [25] could be derived directly from the weak GHR signature scheme.

And, the following result could be derived easily from Theorems 2 and 4. And, security reduction is the same with Theorem 4.

Theorem 8. *The P-WGHR signature scheme is fully-secure.*

7 Conclusion

We showed two new paradigms on how to obtain *fully-secure* signature scheme from any scheme satisfies only a weak security notion called existentially unforgeable against generic chosen message attacks in the standard model. The new paradigms are different from known methods because they are built only on *weakly-secure* signatures, and do not rely on other cryptographic protocols such as one-time signature, or non-standard assumptions such as random oracle model. The transformations are simple, generic, and provably secure in the standard model. The sequential composition method is very efficient in key generation algorithm compared to the second. 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 second paradigm. So, we can use different paradigm in applications according to different requirements.

We also presented several concrete *fully-secure* signature schemes without random oracles converted from two previous *weakly-secure* signature schemes. Their efficiency comparison with the previous secure signatures was also given.

The design of existentially unforgeable secure signature scheme under adaptively chosen message attack, then, can be reduced to that of signature scheme which is secure under only weakly chosen message attack. In the standard model, constructing an efficient signature scheme based on standard assumption is still an open problem. The two new paradigms give one direction in order to solve this problem.

Acknowledgements

This work was partially supported by the 2nd stage of Brain Korea 21 Project sponsored by the Ministry of Education and Human Resources Development, Korea. The third author was supported by the National Natural Science Foundation of China (No. 60773202 and 60633030) and 973 Program (2006CB303104).

References

1. Bellare, M., Micali, S.: How to Sign Given Any Trapdoor Function. J. of the ACM 39, 214–233 (1992)
2. Bellare, M., Rogaway, P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In: ACM CCS 1993, pp. 62–73. ACM Press, New York (1993)
3. Bellare, M., Rogaway, P.: The Exact Security of Digital Signatures-How to Sign with RSA and Rabin. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 399–416. Springer, Heidelberg (1996)
4. Bellare, M., Shoup, S.: Two-Tier Signatures, Strongly Unforgeable Signatures, and Fiat-Shamir without Random Oracles. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 201–216. Springer, Heidelberg (2007)
5. Boneh, D., Boyen, X.: Short Signatures Without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
6. Boneh, D., Lynn, B., Shacham, H.: Short Signatures from The Weil Pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001)
7. Boneh, D., Shen, E., Waters, B.: Strongly Unforgeable Signatures Based on Computational Diffie-Hellman. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T.G. (eds.) PKC 2006. LNCS, vol. 3958, pp. 229–240. Springer, Heidelberg (2006)
8. Camenisch, J., Lysyanskaya, A.: Signature Schemes and Anonymous Credentials from Bilinear Maps. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 56–72. Springer, Heidelberg (2004)
9. Canetti, R., Goldreich, O., Halevi, S.: The Random Oracle Methodology, Revisited. In: STOC 1998, pp. 207–221. ACM, New York (1998)
10. Chevallier-Mames, B., Joye, M.: A Practical and Tightly Secure Signature Scheme without Hash Function. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 339–356. Springer, Heidelberg (2006)
11. Coron, J.-S., Naccache, D.: Security Analysis of The Gennaro-Halevi-Rabin Signature Scheme. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 91–101. Springer, Heidelberg (2000)
12. Cramer, R., Damgård, I.: Secure Signature Schemes Based on Interactive Protocols. In: Coppersmith, D. (ed.) CRYPTO 1995. LNCS, vol. 963, pp. 297–310. Springer, Heidelberg (1995)
13. Cramer, R., Shoup, V.: Signature Schemes Based on the Strong RSA Assumption. ACM TISSEC 3(3), 161–185 (2000); Extended abstract. In: Sixth ACM Conference on Computer and Communication Security (1999)
14. Even, S., Goldreich, O., Micali, S.: On-Line/Off-Line Digital Signatures. Journal of Cryptology 9, 35–67 (1996)
15. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)

16. Gennaro, R., Halevi, S., Rabin, T.: Secure Hash-and-Sign Signatures without The Random Oracle. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 123–139. Springer, Heidelberg (1999)
17. Goh, E.-J., Jarecki, S.: A Signature Scheme as Secure as The Diffie-Hellman Problem. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 401–415. Springer, Heidelberg (2003)
18. Goldwasser, S., Micali, S., Rivest, R.L.: A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. SIAM J. Computing 17(2), 281–308 (1988)
19. Goldwasser, S., Ostrovsky, R.: Invariant Signatures and Non-Interactive Zero-Knowledge Proofs Are Equivalent. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 228–239. Springer, Heidelberg (1993)
20. Huang, Q., Wong, D.S., Zhao, Y.: Generic Transformation to Strongly Unforgeable Signatures. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 1–17. Springer, Heidelberg (2007)
21. Lamport, L.: Constructing Digital Signatures from a One Way Function. Technical Report CSL-98, SRI International (1979)
22. Li, J., Chan, Y.Y., Wang, Y.: A Generic Construction of Secure Signatures Without Random Oracles. In: Gavrilova, M.L., Gervasi, O., Kumar, V., Tan, C.J.K., Taniar, D., Laganá, A., Mun, Y., Choo, H. (eds.) ICCSA 2006. LNCS, vol. 3982, pp. 309–317. Springer, Heidelberg (2006)
23. Lindell, Y.: A Simpler Construction of CCA2-Secure Pulic Key Encryption under General Assumptions. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 241–254. Springer, Heidelberg (2003)
24. Krawczyk, H., Rabin, T.: Chameleon Hashing and Signatures. In: Proc. of NDSS 2000, Internet Society (1998), http://eprint.iacr.org/1998/010
25. Naccache, D., Pointcheval, D., Stern, J.: Twin Signatures: An Alternative to The Hash-and-Sign Paradigm. In: ACM Conference on Computer and Communications Security 2001, pp. 20–27. ACM, New York (2001)
26. Naor, M., Yung, M.: Universal One-Way Hash Functions and Their Cryptographic Applications. In: ACM symposium on Theory of Computing, pp. 33–43. ACM Press, New York (1989)
27. Perrig, A.: The BiBa One-Time Signature and Broadcast Authentication Protocol. In: Eighth ACM Conference on Computer and Communication Security, pp. 28–37. ACM, New York (2001)
28. Pointcheval, D., Stern, J.: Security Arguments for Digital Signatures and Blind Signatures. Journal of Cryptology 13(3), 361–396 (2000)
29. Rivest, R., Shamir, A., Adleman, L.: A Method for Obtaining Digital Signature and Pulbic Key Cryptosystems. Comm. of ACM, 120–126 (1978)
30. Schnorr, C.P.: Efficient Signature Generation by Smart Cards. Journal of Cryptology 4, 161–174 (1991)
31. Shamir, A., Tauman, Y.: Improved Online/Offline Signature Schemes. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 355–367. Springer, Heidelberg (2001)
32. Steinfeld, R., Pieprzyk, J., Wang, H.: How to Strengthen Any Weakly Unforgeable Signature into a Strongly Unforgeable Signature. In: Abe, M. (ed.) CT-RSA 2007. LNCS, vol. 4377, pp. 357–371. Springer, Heidelberg (2006)
33. Waters, B.: Efficient Identity-Based Encryption without Random Oracles. In: Cramer, R.J.F. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
34. Zhang, F., Safavi-Naini, R., Susilo, W.: An Efficient Signature Scheme from Bilinear Pairings and Its Applications. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 277–290. Springer, Heidelberg (2004)

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 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 σ_i=(A_i, B_i, C_i) for $1 \leq i \leq q$. 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 $1 \leq i \leq q$.

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

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 $1 \leq i \leq q$, 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 $1 \leq i \leq q$. $\mathcal{A}'$ 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 $1 \leq i \leq q$. $\mathcal{A}'$ answers the signature query as σ_i=(A_i, B_i, C_i), where A_i=σ'_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 $1 \leq i \leq q$, 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 $1 \leq i \leq q$. $\mathcal{A}'$ 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 σ_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 σ_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 $1 \leq i \leq q$.

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 $prob[\mathrm{E}_1] + prob[\mathrm{E}_2] = prob[\mathcal{A}$ 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 $\frac{1}{2}\cdot$ 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 $\frac{1}{2q}$ $prob[\mathrm{E}_2]$. Therefore, if $\mathcal{A}$ wins with probability ε, the signature scheme Π' with probability at least $\frac{\varepsilon}{2q}$. □

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 σ_i=(A_i, B_i, C_i) for $1 \leq i \leq q$. 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 $1 \leq i \leq q$.

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

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 $m'_1, \cdots, m'_q$ (Assume $\mathcal{A}$ makes at most q queries to signing oracle), and sends the q values m'_i, for $1 \leq i \leq q$, 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 $1 \leq i \leq q$.

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 $1 \leq i \leq q$. $\mathcal{A}'$ 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 σ_i=(A_i, B_i, C_i), where A_i=σ_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 $1 \leq i \leq q$, 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 $m_1'', \cdots, m_q''$ (Assume that $\mathcal{A}$ makes at most q queries to signing oracle), and sends the q values m_i'', for $1 \leq i \leq q$, 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 $\sigma_i' = \mathsf{Sign}'(\overline{sk}, m_i'')$ of the q messages m_i'', with respect to $\overline{pk}$, for $1 \leq i \leq q$.

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 $1 \leq i \leq q$. $\mathcal{A}'$ 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 σ_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 $1 \leq i \leq q$, 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 $\overline{pk}$. $\square$