Divisible On-line/Off-line Signatures

Chong-zhi Gao[1], Baodian Wei[2], and Chunming Tang[3]

[1] College of Computer Science, Guangzhou University ,
Guangzhou 510006, China
czgao@gzhu.edu.cn
[2] Department of Electronics and Communication Engineering,
Sun Yat-sen University Guangzhou, 510275, China
weibd@mail.sysu.edu.cn
[3] Institute of Information Security, Guangzhou University,
Guangzhou 510006, China
tangcm622@hotmail.com

Abstract. On-line/Off-line signatures are used in a particular scenario where the signer must respond quickly once the message to be signed is presented. The idea is to split the signing procedure into two phases. The first phase is off-line: in this phase, the signer does some preparing works before the message to be signed is presented. The second phase is on-line: once the message to be signed is known, the signer utilizes the result of the pre-computation and takes a very short time to accomplish the signing procedure.
In most of these schemes, when signing a message m, a partial signature of m is computed in the off-line phase. We call this part of signature the off-line signature token of message m. In some special applications, the off-line signature tokens might be exposed in the off-line phase. For example, some signers might want to transmit off-line signature tokens in the off-line phase in order to save the on-line transmission bandwidth. Another example is in the case of on-line/off-line threshold signature schemes, where off-line signature tokens are unavoidably exposed to all the users in the off-line phase.
This paper discusses this exposure problem and introduces a new notion: divisible on-line/off-line signatures, in which exposure of off-line signature tokens in off-line phase is allowed. An efficient construction of this type of signatures is also proposed.

Keywords: Signature Schemes, Divisible On-line/Off-line Signatures, Trapdoor Hash Functions, On-line/Off-line Threshold Signatures.

1 Introduction

On-line/Off-line signatures are used in a particular scenario where the signer must respond quickly once the message to be signed is presented. This notion was first introduced by Even, Goldreich and Micali in 1990[11]. The idea of on-line/off-line signatures is to split the signing procedure into two phases. The first phase is off-line: in this phase, the signer does some preparing works before the message to be signed is presented. The second phase is on-line: once the message

to be signed is known, the signer utilizes the result of the pre-computation and uses a very short time to accomplish the signing procedure.

As noted in [11], some signature schemes such as the Fiat-Shamir[12], Schnorr[22], El-Gamal[10] and DSS[18] signature schemes can be naturally viewed as on-line/off-line signature schemes since the first step of these schemes does not depend on the given message, and can thus be carried out off-line. However, these schemes are based on specific security assumptions. For example, the security proofs of the Fiat-Shamir, Schnorr and El-Gamal signatures are based on the random oracle model, and it is not very clear how to analyze the security of the DSS signature scheme[19].

Up to now, there are two general approaches to convert any signature scheme into an on-line/off-line signature scheme. They are Even et al.'s paradigm[11] based on one time signatures and Shamir and Tauman's paradigm[23] based on trapdoor hash functions. Even et al.'s concrete implementation in [11] has a very long signature length and thus is not practical. Shamir-Tauman paradigm greatly reduces the signature length, whilst the on-line computation is fast.

Some recent works in on-line/off-line signatures have been done in [21, 24, 25, 6, 16, 7, 4]. These schemes aim at some specific goals such as improving the efficiency[21], eliminating the random oracle model assumption[16], constructing ID-based schemes[24], constructing threshold schemes[7, 4], avoiding key exposure[6], or avoiding trapdoor hash primitives[25].

Recently in PKC08, Catalano et al.[5] unified Even et al.'s paradigm and Shamir-Tauman paradigm, in the sense that they both use an ordinary signature scheme and a (weak) one time signature scheme as components[4]. Here the trapdoor hash function in Shamir-Tauman paradigm is viewed as a weak one time signature scheme. However, these two paradigms truly have different security characterizations if we consider the partial signature exposure problem described in the next subsection. See next subsection for more details.

1.1 Divisible On-line/Off-line signatures

In most of the on-line/off-line signature schemes([10, 18, 21, 24, 25, 6, 16, 7, 4] and some variations of [12, 22]), when signing a message m, a partial signature of m is computed in the off-line phase. We call this part of signature the off-line signature token of message m. Although the signature generation is broken into two stages, the transmission of a signature is at one time, i.e., the whole signature of a message is transmitted to the recipient at the end of the on-line phase, while nothing is transmitted in the off-line phase.

A question thus naturally arises: can the off-line signature token be transmitted to the recipient off-line? An equivalent question is: is the signature scheme still secure if the adversary is allowed to query the signing oracle with a message depending on this message's off-line signature token? Addressing this question is meaningful because in some special applications, the off-line signature tokens

[4] Here the "weak" means the signature scheme is unforgeable only against generic chosen message attack[15].

might be exposed in the off-line phase. For example, some signers might want to transmit off-line signature tokens in the off-line phase in order to save the on-line transmission bandwidth. Another example is in the case of on-line/off-line threshold signature schemes, where off-line signature tokens are unavoidably exposed to all the users in the off-line phase.

Unfortunately, most on-line/off-line signature schemes can not be proven to be secure if their off-line signature tokens are exposed in the off-line signing phase. In this paper, we introduce a new notion called *divisible on-line/off-line signatures*, in which exposure of off-line signature tokens in off-line singing phase is allowed. To exemplify this new notion, we give in appendix some on-line/off-line signature schemes extracted from existing literatures, which satisfy the new property of divisibility. This paper also presents an efficient construction satisfying the new requirement, which is based on Boneh and Boyen(BB)'s signature scheme[3].

An informal description. Let $\mathcal{OS}$ be an on-line/off-line signature scheme. When signing a message m submitted by a receiver (or generated randomly), the signer uses the signing algorithm of $\mathcal{OS}$ to obtain a signature, say Σ. Informally, we say scheme $\mathcal{OS}$ is divisible if: i) Σ can be separated into two parts Σ^{off} and Σ^{on}, where Σ^{off} is obtained before the message m is known by the signer. ii) Before the signer knows the message, he can send Σ^{off} to the receiver first. In other word, the message requested to be signed in the attack game can depend on the first part of the signature. A formal definition is presented in Section 3.

An on-line/off-line signature scheme is trivially divisible if its Σ^{off} is *null*. For this reason we restrict to non-trivial divisibility in this paper. In the rest of this paper, the word divisible/divisibility usually means a non-trivial case.

Existing Schemes with divisibility. Some existing on-line/off-line signature schemes are listed in Table 1 to show whether they can be proved divisible. We can see that some schemes are divisible such as Scheme Schnorr-OS and Even et al.'s scheme. However, most schemes like Shamir and Tauman's general paradigm can not be proven to have this property.

It is worthwhile noting that Even et al.'s paradigm, which uses an one time signature scheme as a component, is divisible; whereas Shamir and Tauman's general paradigm cannot be proven divisible because it only uses a weak one time signature scheme.

Motivations. Considering the exposure problem might be interesting by itself. Besides, there are two main reasons to consider the divisibility of an on-line/off-line signature scheme:

1. *To save the on-line bandwidth.* If an on-line/off-line scheme is divisible, the signer can send the off-line part of the signature in the off-line phase instead of in the on-line phase. This reduces the on-line bandwidth of the communication channel.

   Remark 1. For example, the signer can pre-compute a series of off-line signature tokens and sends these tokens when the communication channel is not busy. Alternatively, the signer may store these off-line tokens in the form of

Schemes	Divisible?	Note
Fiat-Shamir[12]	No	
Schnorr-OS	Yes	It's a variant of Schnorr signature scheme[22]. See Appendix C.
El-Gamal[10]	No	
DSS[18]	No	
Even et al.'s scheme [11]	Yes	It has a long signature length.
Shamir and Tauman's paradigm (general) [23]	No	Some specific constructions can be proved divisible. See Appendix A,B.
Xu et al.'s scheme [24]	No	It seems divisible. However a deeper analysis shows it is not.
Chen et al.'s scheme [6]	No	
CMTW-OS	Yes	See Appendix A. It is extracted from [7].
BCG-OS	Yes	See Appendix B. It is extracted from [4].

Table 1. Some on-line/off-line signature schemes. The second column shows whether they can be proved to be divisible.

a DVD/CD and send the disk to the receiver directly since these off-line tokens do not depend on the messages to be later signed. At the same time, to ensure the one-to-one correspondence of the off-line tokens with the on-line ones, we can append a digital label to each off-line/on-line signature token. Note that doing this does not much increase the on-line signature length. For example, labels of 15 bit each can distinguish more than 32,000 signature tokens whereas each on-line signature token is at least 160 bits up to date. Furthermore, the index in the labels can be reused while the unused off-line signature tokens are exhausted.

2. *To construct on-line/off-line threshold signatures.* An on-line/off-line threshold signature scheme[7, 4] is a threshold signature scheme[8, 9] which can be partitioned into off-line and on-line phases. There are two main approaches to prove the unforgeability of a threshold signature scheme: the direct reduction approach (e.g., the part of reduction to the one-more-discrete-log assumption of Theorem 2 in [7], and the part of reduction to the discrete log assumption of Theorem 1 in [4]) and the simulation approach(e.g., [13, 20, 14]). In the direct reduction approach, the security of a threshold signature scheme is directly reduced to the hardness of an underlying hard problem such as the factoring problem or the discrete log problem. In the simulation approach, the security of a threshold signature scheme is reduced to the unforgeability of its basic signature scheme (This reduction approach is called *simulation*, and the property which guarantees the success of simulation is called the *simulatability* of a threshold signature scheme). In essence, the two approaches are the same, in the sense that the security is reduced to the hardness of an underlying hard problem in the end. However, if the basic signature scheme is known to be unforgeable, the simulation approach will simplify the proof.

When proving the unforgeability of an on-line/off-line threshold signature scheme $\mathcal{OTS}$ in the simulation approach, one must be careful that $\mathcal{OTS}$ should be proved simulatable to a divisible on-line/off-line signature scheme, i.e., the simulation of the off-line signing phase should not depend on the message to be signed in the on-line phase. The reason is that in the attack game of on-line/off-line threshold signatures, the adversary can adaptively choose messages to be signed depending on his off-line signing view. Thus, a divisible on-line/off-line signature is a key component of an on-line/off-line threshold signature.

Related work. The notion of divisible on-line/off-line signatures is first explicitly given in this paper, but the original idea goes back to [7, 4]. When proving the unforgeability of an on-line/off-line threshold signature scheme, the authors noticed that the off-line simulation of the scheme should not depend on the message to be signed. Thus, the on-line/off-line signature schemes(CMTW-OS and BCG-OS, see Appendix A,B) extracted from [7, 4] are divisible. Besides, Even et al.'s paradigm, which utilizes an one time signature scheme, is also divisible. This work is before [7, 4].

Scheme CMTW-OS and BCG-OS are both based on Shamir and Tauman's hash-sign-switch paradigm[23]. But Shamir-Tauman paradigm itself cannot be proven divisible. However, if the specific trapdoor hash functions (e.g., the trapdoor hash functions in CMTW-OS and BCG-OS) used in Shamir-Tauman paradigm can be viewed as a fully secure one time signature scheme, Shamir-Tauman paradigm can be unified again into Even et al.'s general paradigm, in the sense that these two paradigms both uses an one time signature scheme as a component and thus can be proven divisible[5]. This is exactly the reason that Scheme CMTW-OS and BCG-OS can be proven divisible.

Although Even et al.'s general paradigm (especially its practical implementations CMTW-OS and BCG-OS) can convert any ordinary signature scheme into a divisible on-line/off-line signature scheme, it has two drawbacks: i)A basic signature scheme is needed as a component. Thus, compared to our proposed scheme whose signing algorithm requires essentially only one exponential computation, these schemes requires an additional standard signature computation. ii)In addition to a basic assumption such as the discrete log assumption, the security of these schemes also depends on the security of the basic signature scheme. See Section 4.3 for more details.

1.2 Our Contribution

In this paper, we first explicitly give the notion of divisible on-line/off-line signatures. To illustrate this notion, we show in Appendix C how to prove Schnorr signature scheme[22] is divisible. Furthermore, using the double trapdoor technique, we present an efficient divisible scheme, which is based on BB signature scheme[3]. In our new scheme, the use of trapdoor hash function is eliminated and thus our scheme is more efficient than divisible schemes in [7, 4]. (In fact,

a trapdoor hash function is implicitly integrated into the new scheme. It can be explicitly seen in the output stage of the proof of Theorem 1.)

To sum up, our divisible scheme has the following advantages:

1. Its security is proven in the standard model (instead of in the random oracle model).
2. Its computational cost of on-line signing is only one modular multiplication. This is comparable to the state-of-the-art on-line/off-line signature schemes.
3. Its overall computational cost of singing is essentially only one scalar exponentiation in a bilinear group. [5] This is superior to other divisible on-line/off-line signature schemes whose unforgeability is proved in the standard model.

Finally, an application to on-line/off-line threshold signatures is presented. We show that based on a divisible on-line/off-line signature scheme, an on-line/off-line threshold signature scheme can be proven secure if it is simulatable.

1.3 Organization

The rest of this paper is organized as follows. In Section 2, we give some preliminaries. Section 3 gives the security model of divisible on-line/off-line signatures. Section 4 presents an efficient divisible on-line/off-line signature scheme whose security is proven in the standard model. An application to on-line/off-line threshold signature schemes is introduced in Section 5.

2 Preliminaries

2.1 Notations and Definitions

We denote by $\mathbb{N}$ the set of natural numbers, and by $\mathbb{Z}$ the set of integers. If $k \in \mathbb{N}$, we denote by 1^k the concatenation of k ones and by $\{0,1\}^k$ the set of bitstrings of bitlength k. By $\{0,1\}^*$, we denote the set of bitstrings of arbitrary bitlength.

If S is a set, then the notation $x \xleftarrow{\mathrm{R}} S$ denotes that x is selected randomly from the set S. Similarly, $x \in_R S$ denotes x is a random element of S. If $\mathcal{A}$ is an algorithm, by $\mathcal{A}(\cdot)$ we denote that $\mathcal{A}$ receives only one input. If $\mathcal{A}$ receives two inputs we write $\mathcal{A}(\cdot,\cdot)$ and so on. If $\mathcal{A}(\cdot)$ is a probabilistic algorithm, $y \leftarrow \mathcal{A}^{\mathcal{O}_1,\mathcal{O}_2,\cdots}(x_1, x_2, \dots)$ means that on input $x_1, x_2, \dots$ and with access to oracles $\mathcal{O}_1, \mathcal{O}_2, \dots$, $\mathcal{A}$'s output is y. If $p(\cdot,\cdot,\dots)$ is a predicate, the notation $\Pr[p(x,y,\dots) : x \xleftarrow{\mathrm{R}} S; y \xleftarrow{\mathrm{R}} T; \dots]$ denotes the probability that $p(x,y,\dots)$ will be true after the ordered execution of the algorithms $x \xleftarrow{\mathrm{R}} S, y \xleftarrow{\mathrm{R}} T, \dots$. "PPT" is an abbreviation for "probabilistic polynomial-time".

Definition 1 (Negligible Function). *A function $\epsilon : \mathbb{N} \to \mathbb{R}$ is negligible if for all $c > 0$, $\epsilon(k) < 1/k^c$ for all sufficiently large k.*

[5] "Scalar exponentiation" in this context is the repeated application of a group operation to the same element. In some contexts, it is also called scalar multiplication.

Definition 2 (Discrete Logarithm Assumption). *Let $p \in \{0,1\}^k$ be a prime. Let $\mathbb{G}$ be a group of order p and let $g \in \mathbb{G}$ be a generator of $\mathbb{G}$. Solving the discrete logarithm problem in $\mathbb{G}$ is to compute x, given $h = g^x \in \mathbb{G}$ where x is randomly selected in $\mathbb{Z}_p$. The discrete logarithm assumption in $\mathbb{G}$ states that the discrete logarithm problem is hard to solve, i.e., for any PPT algorithm $\mathcal{A}$, the following probability is negligible in k.*

$$\epsilon(k) = \Pr[\mathcal{A}(\mathsf{descr}(\mathbb{G}), g, h) = x : x \overset{\mathrm{R}}{\leftarrow} \mathbb{Z}_p; h \leftarrow g^x]$$

where $\mathsf{descr}(\mathbb{G})$ is a description of $\mathbb{G}$ which contains the value of p and other group parameters.

Definition 3 (One-More-Discrete-Log Assumption[1]). *Let $p \in \{0,1\}^k$ be a prime. Let $\mathbb{G}$ be a group of order p and let $g \in \mathbb{G}$ be a generator of $\mathbb{G}$. Define $DL_g(\cdot)$ as a oracle which on input $h \in \mathbb{G}$, returns $\mathrm{dl}_g h \in \mathbb{Z}_p$ (the discrete logarithm of h to the base of g).*

Solving the n-DL problem is to compute $x_1, x_2, \ldots, x_{n+1}$, with access to the oracle $DL_g(\cdot)$ at most n times, given $h_1 = g^{x_1}, \ldots, h_{n+1} = g^{x_{n+1}} \in \mathbb{G}$ where $x_i (1 \leq i \leq n+1)$ are randomly selected in $\mathbb{Z}_p$. The one-more-discrete-log assumption in $\mathbb{G}$ states that n-DL problem is hard to solve for any $n \in \mathbb{N}$, i.e., for any $n \in \mathbb{N}$ and any PPT algorithm $\mathcal{A}$ with access to oracle $DL_g(\cdot)$ at most n times, the following probability is negligible in k.

$$\begin{aligned}\epsilon(k) = \Pr[\mathcal{A}^{DL_g(\cdot)} \ & (\mathsf{descr}(\mathbb{G}), g, h_1, \ldots, h_{n+1}) = (x_1, \ldots, x_{n+1}) \\ & : x_1 \overset{\mathrm{R}}{\leftarrow} \mathbb{Z}_p, \ldots, x_{n+1} \overset{\mathrm{R}}{\leftarrow} \mathbb{Z}_p; h_1 \leftarrow g^{x_1}, \ldots, h_{n+1} \leftarrow g^{x_{n+1}}]\end{aligned}$$

where $\mathsf{descr}(\mathbb{G})$ is a description of $\mathbb{G}$ which contains the value of p and other group parameters.

Definition 4 (Bilinear Paring). *Let $\mathbb{G}, \mathbb{G}_T$ be two multiplicative cyclic group of prime order p. A bilinear pairing on $(\mathbb{G}, \mathbb{G}_T)$ is a function $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ which has the following properties:*

1. *Bilinear: $e(u^a, v^b) = e(u, v)^{ab}$, for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}$.*
2. *Non-degenerate: $e(u, v) \neq 1$ for all $u, v \in \mathbb{G}$.*
3. *Computable: paring $e(u, v)$ can be efficiently computed for all $u, v \in \mathbb{G}$.*

For generality, one can set $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ where $\mathbb{G}_1 \neq \mathbb{G}_2$. An efficiently computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ can convert the security proof of the simple case where $\mathbb{G}_1 = \mathbb{G}_2$ to the general case.

Definition 5 (q-SDH Assumption[3]). *Let $\mathbb{G}$ be a group of prime order p. Let g be a generator of $\mathbb{G}$. Solving the q-SDH problem in $\mathbb{G}$ is to compute a pair $(c, g^{1/(x+c)})$ where $c \in \mathbb{Z}_p^*$, given a $(q+1)$-tuple $(g, g^x, g^{(x^2)}, \ldots, g^{(x^q)})$. The q-SDH assumption in group $\mathbb{G}$ states that the q-SDH problem in $\mathbb{G}$ is hard to solve, i.e., for any PPT algorithm $\mathcal{A}$, the following probability is negligible in k.*

$$\epsilon(k) = \Pr[\mathcal{A}(g, g^x, g^{(x^2)}, \ldots, g^{(x^q)}) = (c, g^{1/(x+c)}) : x \overset{\mathrm{R}}{\leftarrow} \mathbb{Z}_p^*]$$

The following lemma states that given a q-SDH problem instance $(g, g^x, \ldots, g^{(x^q)})$, we can construct a new 1-SDH prolem instance (h, h^x) with $q-1$ known solutions $(c_i, s_i = h^{1/(x+c_i)})$ where any new solution reveals a solution to the original problem instance. We refer the reader to the Lemma 3.2 of [3] for the proof of this lemma.

Lemma 1. *There exists a PPT algorithm Γ which satisfies:*

- *Its inputs are:*
  1. $\mathsf{descr}(\mathbb{G})$. *A description of a group $\mathbb{G}$ with prime order p.*
  2. $(g, g^x, \ldots, g^{(x^q)})$. *A q-SDH problem instance in Group $\mathbb{G}$ where $q \in \mathbb{N}$.*
  3. $c_1, \ldots, c_{q-1} \in \mathbb{Z}_p^*$ *where all* $c_i \not\equiv -x \pmod{p}$.
- *It outputs a PPT algorithm Δ and a tuple $((h, u), (s_1, \ldots, s_{q-1})) \in \mathbb{G}^2 \times \mathbb{G}^{q-1}$ which satisfy:*
  1. $u = h^x$.
  2. $s_i = h^{1/(x+c_i)}$, *i.e.,* (c_i, s_i) $(1 \leq i \leq q-1)$ *are solutions of the 1-SDH problem instance (h, h^x).*
  3. *By using Algorithm Δ, any new solution $(c^*, s^*) \neq (c_i, s_i)$ for the 1-SDH problem instance (h, h^x) reveals a solution to the original instance,i.e., on inputs (c^*, s^*) where $(c^*, s^*) \neq (c_i, s_i)$ for all $i \in \{1, \ldots, q-1\}$ and $s^* = h^{1/(x+c^*)}$, Δ can output a pair $(c, g^{1/(x+c)}) \in \mathbb{Z}_p^* \times \mathbb{G}$ in polynomial time.*

3 Security Model

We give the security model of divisible online/offline signatures and some security notions.

3.1 Syntax

A divisible online/offline signature scheme ($\mathcal{DOS}$) is a tuple of algorithms (KeyGen, $\mathsf{Sign}^{\mathsf{off}}, \mathsf{Sign}^{\mathsf{on}}, \mathsf{Ver}$).

- $(pk, sk) \leftarrow \mathsf{KeyGen}(1^k)$. The Key generation algorithm,a PPT algorithm which on input a security parameter $k \in \mathbb{N}$, outputs a public/private key pair (pk, sk).
- $(\Sigma_i^{\mathrm{off}}, St_i) \leftarrow \mathsf{Sign}^{\mathsf{off}}(sk)$. The i-th ($i \in \mathbb{N}$) executing of the off-line signing algorithm, a PPT algorithm which on input a private key, outputs a (public) off-line signature token Σ_i^{off} and a (secret) state information St_i. The state information is kept secret and will be transmitted to the i-th executing of the on-line signing algorithm.
- $\Sigma_i^{\mathrm{on}} \leftarrow \mathsf{Sign}^{\mathsf{on}}(sk, St_i, m_i)$. The i-th ($i \in \mathbb{N}$) executing of the on-line signing algorithm, a PPT algorithm which on input sk, a state information St_i and a message m_i, outputs an on-line signature token Σ_i^{on}. The signature for m_i is defined as $\Sigma_i = (\Sigma_i^{\mathrm{off}}, \Sigma_i^{\mathrm{on}})$
- $0/1 \leftarrow \mathsf{Ver}(pk, m, \Sigma)$. The verification algorithm, a PPT algorithm which on input the public key pk, a message m and a signature Σ, outputs 0 or 1 for reject or accept respectively.

Completeness: It is required that if $(\Sigma^{\text{off}}, St) \leftarrow \mathsf{Sign}^{\text{off}}(sk)$ and $\Sigma^{\text{on}} \leftarrow \mathsf{Sign}^{\text{on}}(sk, St, m)$, then $\mathsf{Ver}(pk, m, \Sigma) = 1$ for all (pk, sk) generated by $\mathsf{KeyGen}(1^k)$.

3.2 Security Notation

In the following, we define a security notion for a divisible on-line/off-line signature scheme.

EU-CMA: For a divisible on-line/off-line signature scheme $\mathcal{DOS}$, existential unforgeability against adaptive chosen message attacks is defined in the following game. This game is carried out between a challenger $\mathcal{S}$ and an adversary $\mathcal{A}$. The adversary $\mathcal{A}$ is allowed to make queries to an off-line signing oracle $\mathsf{Sign}^{\text{off}}(sk)$ and an on-line signing oracle $\mathsf{Sign}^{\text{on}}(sk, St, \cdot)$ defined in Section 3.1. We assume that if $\mathcal{A}$ makes the i-th on-line signature query then it has already made the i-th off-line signature query. This requirement is reasonable since the signer always execute his i-th off-line signature signing before his i-th on-line signing. If in the random oracle model, $\mathcal{A}$ is also allowed to make queries to a hash oracle $h(\cdot)$ which on input a message in $\{0,1\}^*$, outputs a hash value of this message. The attack game is as follows:

1. The challenger runs KeyGen on input 1^k to get (pk, sk). pk is sent to $\mathcal{A}$.
2. On input $(1^k, pk)$, $\mathcal{A}$ is allowed to query the oracles $\mathsf{Sign}^{\text{off}}(sk)$, $\mathsf{Sign}^{\text{on}}(sk, St, \cdot)$ (and $h(\cdot)$ if in the random oracle model) polynomial times. The i-th state information St_i is transmitted from the i-th executing of $\mathsf{Sign}^{\text{off}}(sk)$.
3. $\mathcal{A}$ outputs a pair (m, Σ).

The adversary wins the game if the message m has never been queried to the on-line signing oracle $\mathsf{Sign}^{\text{on}}(sk, St, \cdot)$ and $\mathsf{Ver}(pk, m, \Sigma) = 1$ holds. Let $\text{Adv}_{\mathcal{A},\mathcal{DOS}}$ be the *advantage* of the adversary $\mathcal{A}$ in breaking the signature scheme, i.e.,

$$\begin{aligned}\text{Adv}_{\mathcal{A},\mathcal{DOS}} =& \Pr[\mathsf{Ver}(pk, m, \Sigma) = 1 : \\ & (pk, sk) \leftarrow \mathsf{KeyGen}(1^k); (m, \Sigma) \leftarrow \mathcal{A}^{\mathsf{Sign}^{\text{off}}(sk), \mathsf{Sign}^{\text{on}}(sk, St, \cdot), \eta(\cdot)}]\end{aligned}$$

where

$$\eta(\cdot) = \begin{cases} null & \text{(in the standard model)} \\ h(\cdot) & \text{(in the random oracle model)} \end{cases}$$

and $\mathcal{A}$ has never requested the signature of m from the on-line signing oracle. The probability is taken over the internal coin tosses of the algorithm KenGen and $\mathcal{A}$.

Definition 6. *An adversary $\mathcal{A}$ $(t, q_{\text{off}}, q_{\text{on}}, \epsilon)$-breaks a divisible online/offline signature scheme $\mathcal{DOS}$ in the standard model if $\mathcal{A}$ runs in time at most t, makes at most q_{off} queries to the off-line signing oracle, at most q_{on} queries to the on-line signing oracle, and $\text{Adv}_{\mathcal{A},\mathcal{DOS}}$ is at least ϵ.*

An adversary $\mathcal{A}$ $(t, q_{\text{off}}, q_{\text{on}}, q_{\text{h}}, \epsilon)$*-breaks a divisible online/offline signature scheme* $\mathcal{DOS}$ *in the random oracle model if* $\mathcal{A}$ *runs in time at most* t*, makes at most* q_{off} *queries to the off-line signing oracle, at most* q_{on} *queries to the on-line signing oracle, at most* q_{h} *queries to the hash oracle, and* $\text{Adv}_{\mathcal{A},\mathcal{DOS}}$ *is at least* ϵ*.*

A divisible on-line/off-line signature scheme $\mathcal{DOS}$ *is existentially unforgeable under adaptive chosen message attacks if for every PPT adversary* $\mathcal{A}$*,* $\text{Adv}_{\mathcal{A},\mathcal{DOS}}$ *is negligible.*

Difference to the previous definition. The above definition is different from the attack game for an ordinary oo signature scheme where the adversary is only allowed to query the oracle $\mathsf{Sign}(sk, \cdot)$ (and a hash oracle if in the random oracle). In other word, in the attack game of an ordinary scheme, the off-line signature token is returned to the adversary only *after* the message to be signed is submitted, whilst in the game for a divisible scheme, the adversary obtains the off-line signature token of a message *before* he submits this message.

Thus, the unforgeability defined above is stronger than the unforgeability defined as usual for ordinary on-line/off-line signatures. Note, however, that the unforgeability defined as usual is enough for the applications where off-line signature tokens are not exposed in the off-line signing phase.

4 A Divisible On-line/Off-line Signature Scheme Based on The *q*-SDH Assumption

In this section, we propose an efficient divisible on-line/off-line signature scheme whose security is proven in the standard model. This scheme is based on Boneh and Boyen's signature scheme[3].

4.1 Construction

Let $\mathbb{G}$ be a bilinear group of prime order p. Assume the message space is $\mathbb{Z}_p^*$. Note that using a collision resistant hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$, one can extend the message domain to $\{0,1\}^*$. The new divisible oo signature scheme $\mathsf{SDH\text{-}OS} = (\mathsf{KeyGen}, \mathsf{Sign}^{\mathsf{off}}, \mathsf{Sign}^{\mathsf{on}}, \mathsf{Ver})$, where

- KeyGen. Pick a random generator $g \in \mathbb{G}$. Choose random $x, y, z \in_R \mathbb{Z}_p^*$, and compute $X = g^x \in \mathbb{G}$, $Y = g^y \in \mathbb{G}$ and $Z = g^z \in \mathbb{G}$. Also compute $v = e(g, g) \in \mathbb{G}_T$. The public key is (g, X, Y, Z, v). The private key is (x, y, z).
- $\mathsf{Sign}^{\mathsf{off}}$. (The i-th run). Choose a random $\theta \in \mathbb{Z}_p^*$ such that $x + \theta \neq 0$. Compute $\sigma = g^{\frac{1}{(x+\theta)}}$ where $\frac{1}{(x+\theta)}$ is the inverse of $(x + \theta)$ in $\mathbb{Z}_p^*$. Store the state information θ. The off-line signature token is σ.
- $\mathsf{Sign}^{\mathsf{on}}$. (The i-th run, on a message m). Retrieve from the memory the i-th state information θ. Compute $r, w \in \mathbb{Z}_p^*$ such that:

$$m + yr + zw = \theta.$$

This can be done by first selecting a random $r \in \mathbb{Z}_p^*$ such that $m + yr \neq \theta$, and computing $w = (\theta - m - yr)z^{-1} \mod p$. The on-line signature token is (r, w)

– Ver. Given a message $m \in \mathbb{Z}_p^*$ and a signature (σ, r, w), verify that whether $e(\sigma, Xg^m Y^r Z^w) = v$.

Remark 2. To reduce the on-line signing cost, we can move the selection of r and computing $y \cdot r$ to the off-line phase. Thus, the on-line signing requires only 1 modular multiplication in $\mathbb{Z}_p$.

Completeness: Note that

$$\begin{aligned} e(\sigma, Xg^m Y^r Z^w) &= e(g^{1/(x+\theta)}, g^{x+m+yr+zw}) \\ &= e(g^{1/(x+\theta)}, g^{x+\theta}) \\ &= e(g, g) = v \end{aligned}$$

Thus the proposed scheme satisfies the property of completeness.

4.2 Security

Theorem 1. *The divisible on-line/off-line signature scheme* SDH-OS *is existentially unforgeable under adaptive chosen message attacks, provided that the q-SDH assumption holds in Group* $\mathbb{G}$.

Proof. We prove this theorem by contradiction. Assume there exists an Algorithm $\mathcal{A}$ which $(t, q_{\text{off}}, q-1, \epsilon)$ breaks the unforgeability of SDH-OS in the game defined in Section 3. Then we construct an Algorithm $\mathcal{B}$ which breaks the q-SDH problem in polynomial time with probability $\epsilon' \geq \frac{\epsilon}{3} \cdot (1 - \frac{q}{p})$. In the proof, the multi-mode technique of [3] is used.

Let $\mathbb{G}$ be a group of prime order p. Let g be a generator of $\mathbb{G}$. Algorithm $\mathcal{B}$ is given a q-SDH problem instance $(g, g^\tau, g^{(\tau^2)}, \ldots, g^{(\tau^q)})$. To solve this problem instance, $\mathcal{B}$ works as follows.

Setup: Algorithm $\mathcal{B}$ selects $c_{\text{mode}} \in_R \{1, 2, 3\}$ and a list of messages $c_1, \ldots, c_{q-1} \in_R \mathbb{Z}_p^*$. We may assume $c_i + \tau \neq 0$ for all $i \in \{1, \ldots, q-1\}$, or else $\mathcal{B}$ has already obtained τ and thus the q-SDH problem is solved. Next, $\mathcal{B}$ feeds Algorithm Γ in Lemma 1 with inputs $\mathsf{descr}(\mathbb{G}), (g, g^\tau, g^{(\tau^2)}, \ldots, g^{(\tau^q)}), (c_1, \ldots, c_{q-1})$ to get an algorithm Δ and $((h, u), (s_1, \ldots, s_{q-1})) \in \mathbb{G}^2 \times \mathbb{G}^{q-1}$. Note that as described in Lemma 1, $u = h^\tau$ and $s_i = h^{1/(\tau + c_i)}$ hold. Algorithm $\mathcal{B}$ computes $v = e(h, h)$ and proceeds as follows.

- If $c_{\text{mode}} = 1$, selects $y, z \in_R \mathbb{Z}_p^*$ and sends to $\mathcal{A}$ a public key (h, u, h^y, h^z, v).
- If $c_{\text{mode}} = 2$, selects $x, z \in_R \mathbb{Z}_p^*$ and sends to $\mathcal{A}$ a public key (h, h^x, u, h^z, v).
- If $c_{\text{mode}} = 3$, selects $x, y \in_R \mathbb{Z}_p^*$ and sends to $\mathcal{A}$ a public key (h, h^x, h^y, u, v).

In either case, we denote by (h, X, Y, Z, v) the public key that $\mathcal{A}$ received.

Simulating the Signing Oralce (Off-line): Algorithm $\mathcal{A}$ can submit at most q_{off} off-line signing queries where $q_{\text{off}} \geq q - 1$. Then, upon the i-th $(1 \leq i \leq q-1)$ query,

- If $c_{\text{mode}} = 1$, $\mathcal{B}$ returns $\sigma_i = s_i$ as the i-th off-line signature token.
- If $c_{\text{mode}} = 2$, $\mathcal{B}$ selects $r_i \in_R \mathbb{Z}_p^*$, and returns $\sigma_i = (s_i)^{\frac{1}{r_i}}$ as the i-th off-line signature token.
- If $c_{\text{mode}} = 3$, $\mathcal{B}$ selects $w_i \in_R \mathbb{Z}_p^*$, and returns $\sigma_i = (s_i)^{\frac{1}{w_i}}$ as the i-th off-line signature token.

Upon the i-th ($q \leq i \leq q_{\text{off}}$) query, $\mathcal{B}$ just returns a random element in $\mathbb{G}$.

Simulating the Signing Oralce (On-line): Algorithm $\mathcal{B}$ creates a H-list of 4-tuples which is initialized to empty. Algorithm $\mathcal{A}$ can submit at most $q-1$ on-line signing queries. Upon the i-th query input m_i,

- If $c_{\text{mode}} = 1$, $\mathcal{B}$ selects $r_i \in_R \mathbb{Z}_p^*$, and sets $w_i = (c_i - m_i - yr_i)z^{-1} \mod p$. If $w_i = 0$, Algorithm $\mathcal{B}$ reports failure and aborts. Otherwise $\mathcal{B}$ outputs (r_i, w_i) as the i-th online signature token.
- If $c_{\text{mode}} = 2$, $\mathcal{B}$ selects $r_i \in_R \mathbb{Z}_p^*$, and sets $w_i = (c_i r_i - x - m_i)z^{-1} \mod p$. If $w_i = 0$, Algorithm $\mathcal{B}$ reports failure and aborts. Otherwise $\mathcal{B}$ outputs (r_i, w_i) as the i-th online signature token.
- If $c_{\text{mode}} = 3$, $\mathcal{B}$ selects $w_i \in_R \mathbb{Z}_p^*$, and sets $r_i = (c_i w_i - x - m_i)y^{-1} \mod p$. If $r_i = 0$, Algorithm $\mathcal{B}$ reports failure and aborts. Otherwise $\mathcal{B}$ outputs (r_i, w_i) as the i-th online signature token.

It can be verified that in either case $\mathcal{B}$ outputs a valid signature (σ_i, r_i, w_i) on the message m_i if $\mathcal{B}$ does not abort. In either case, Algorithm $\mathcal{B}$ sets $W_i = h^{m_i} Y^{r_i} Z^{w_i}$ and adds the 4-tuple (m_i, r_i, w_i, W_i) to the H-list.

Output: It can be shown that the simulated off-line/on-line singing oracles are indistinguishable from the real ones for $\mathcal{A}$ if $\mathcal{B}$ does not abort. Thus, with probability ϵ, Algorithm $\mathcal{A}$ outputs a valid forgery $(m_*, \sigma_*, r_*, w_*)$ where $m_* \neq m_i$ for all $i \in \{1, \ldots, q-1\}$. Algorithm $\mathcal{B}$ computes $W_* = h^{m_*} Y^{r_*} Z^{w_*}$ and searches the H-list for a tuple of the form $(\cdot, \cdot, \cdot, W_*)$. One of the following events must happen:

Event 1: $W_* \neq W_i$ for all $i \in \{1, \ldots, q-1\}$. Let $b_{\text{type}} = 1$ for this case.
Event 2: $W_* = W_i$ for some $i \in \{1, \ldots, q-1\}$, and $r_* \neq r_i$. Let $b_{\text{type}} = 2$ for this case.
Event 3: $W_* = W_i$ for some $i \in \{1, \ldots, q-1\}$, and $r_* = r_i$, but $w_* \neq w_i$. Let $b_{\text{type}} = 3$ for this case.

Algorithm $\mathcal{B}$ aborts if $b_{\text{type}} \neq c_{\text{mode}}$. Otherwise,

- If $c_{\text{mode}} = b_{\text{type}} = 1$, then let $c_* = m_* + yr_* + zw_*$. We can see (c_*, σ_*) is a new solution to the 1-SDH problem instance (h, h^τ) since from the verification equation we can get $\sigma_* = h^{1/(\tau + c_*)}$ and from $W_* \neq W_i$ for all i we can get $c_* \neq c_i$ for all i. From Lemma 1, the original q-SDH problem instance can be solved in polynomial time by Algorithm Δ.
- If $c_{\text{mode}} = b_{\text{type}} = 2$, then for some i, $h^{m_*} Y^{r_*} Z^{w_*} = h^{m_i} Y^{r_i} Z^{w_i}$ and $r_* \neq r_i$ hold. Thus, Algorithm $\mathcal{B}$ can get $\tau = \text{dl}_h u = \text{dl}_h Y = [(w_i - w_*)z + (m_i - m_*)] \cdot (r_* - r_i)^{-1} \mod p$.

- If $c_{\text{mode}} = b_{\text{type}} = 3$, then for some i, $h^{m_*}Y^{r_*}Z^{w_*} = h^{m_i}Y^{r_i}Z^{w_i}$ and $w_* \neq w_i$ hold. Thus, Algorithm $\mathcal{B}$ can get $\tau = \text{dl}_h u = \text{dl}_h Z = [(r_i - r_*)y + (m_i - m_*)] \cdot (w_* - w_i)^{-1} \mod p$.

Thus, if $\mathcal{B}$ does not abort and $b_{\text{type}} = c_{\text{mode}}$ holds, then $\mathcal{B}$ successfully solves the original q-SDH problem instance.

Remark 3. Here $W = h^m Y^r Z^w$ plays a role of "trapdoor hash function" in the scheme.

Next, let's analyze the probability that $\mathcal{B}$ succeeds. $\mathcal{B}$ succeeds if the following three events happen:

Event A: $\mathcal{B}$ does not abort in the simulating stage.
Event B: $\mathcal{A}$ successfully outputs a forgery in its own challenge game.
Event C: $\mathcal{B}$ does not abort in the output stage, i.e., $b_{\text{type}} = c_{\text{mode}}$ holds.

In the i-th round on-line signing simulation, Algorithm $\mathcal{B}$ aborts if i) r_i is chosen to be $(c_i - m_i)y^{-1}$ when $c_{\text{mode}} = 1$, or ii) r_i is chosen to be $(x + m_i)c_i^{-1}$ when $c_{\text{mode}} = 2$, or iii) w_i is chosen to be $(x + m_i)c_i^{-1}$ when $c_{\text{mode}} = 3$. This happens with probability $1/(p-1)$ in each round. Thus, in the simulation stage, $\mathcal{B}$ does not abort with probability at least $1-(q-1)/(p-1)$, i.e., $\Pr[A] = 1-(q-1)/(p-1)$. Since $\mathcal{A}$ outputs a forgery with probability ϵ if $\mathcal{B}$ does not abort in the simulation stage, we have that $\Pr[B|A] = \epsilon$. Since the value of c_{mode} is independent of b_{type}, we have that $\Pr[C|AB] = 1/3$. So,

$$\begin{aligned}\Pr[ABC] &= \Pr[C|AB] \cdot \Pr[B|A] \cdot \Pr[A] \\ &\geq \frac{1}{3} \cdot \epsilon(1 - \frac{q-1}{p-1}) \\ &\geq \frac{\epsilon}{3} \cdot (1 - \frac{q}{p})\end{aligned}$$

We have constructed a PPT algorithm which breaks the q-SDH problem with non-negligible probability. This contradicts the q-SDH assumption and thus the theorem is proved. ■

4.3 Comparison and Discussion

We compare our new scheme SDH-OS with some known divisible on-line/off-line signature schemes in Table 2. To achieve the same secuirty level, we assume the parameter p in our new scheme and Schemes CMTW-OS, BCG-OS and Schnorr-OS are all k-bit long. When using an elliptic curves with $k = 160$, our scheme has the same security level with a 1024-bit key RSA signature[3]. In this case, our scheme has a 160-bit off-line signature length and a 320-bit on-line signature length. In comparison, we omit additions in the signing algorithm.

To our knowledge, the most efficient divisible on-line/off-line signature scheme is Scheme Schnorr-OS. However, its security proof is based on the random oracle model(ROM). Our new scheme's security is proven in the standard model. Its

overall computational cost of signing is only one scalar exponentiation in the group $\mathbb{G}$ (i.e., roughly k squarings and $k/2$ multiplications in $\mathbb{G}$), which is comparable to Scheme Schnorr-OS and is superior to other schemes whose security is proved in the standard model(See Remark 4). Our new scheme's on-line signing requires only 1 modular multiplication in $\mathbb{Z}_p$. This is very efficient and comparable to other efficient schemes. The main drawback of our scheme is the on-line signature length is $2\log_2 p$, which is twice the length of Scheme CMTW-OS or Schnorr-OS. Thus, it remains an unsolved problem to find a divisible scheme whose security is proven in the standard model and whose performance is comparable to Scheme Schnorr-OS.

Schemes	$\mathsf{Sign}^{\mathsf{off}}$	$\mathsf{Sign}^{\mathsf{on}}$	Ver	Signature Size Off-line/ On-line	Assumptions
New Scheme	k sq. in $\mathbb{G}$ $\frac{k}{2}$ mult. in $\mathbb{G}$	1 mult. in $\mathbb{Z}_p$	k sq. in $\mathbb{G}$ $\frac{7}{8}k$ mult. in $\mathbb{G}$ 1 pairing	k bits / $2k$ bits	q-SDH
CMTW-OS (Appendix A)	k sq. in $\mathbb{G}$ $\frac{3}{4}k$ mult. in $\mathbb{G}$ 1 stand. sig	1 mult. in $\mathbb{Z}_p$	k sq. in $\mathbb{G}$ $\frac{3}{4}k$ mult. in $\mathbb{G}$ 1 stand. ver	1 stand.sig / k bits	Sig, one-more-discrete-log
BCG-OS (Appendix B)	k sq. in $\mathbb{G}$ $\frac{7}{8}k$ mult. in $\mathbb{G}$ 1 stand. sig	1 mult. in $\mathbb{Z}_p$	k sq. in $\mathbb{G}$ $\frac{7}{8}k$ mult. in $\mathbb{G}$ 1 stand. ver	1 stand.sig / $2k$ bits	Sig, discrete log
Schnorr-OS (Appendix C)	k sq. in $\mathbb{G}$ $\frac{k}{2}$ mult. in $\mathbb{G}$	1 mult. in $\mathbb{Z}_p$	k sq. in $\mathbb{G}$ $\frac{3}{4}k$ mult. in $\mathbb{G}$	k bits/ k bits	ROM, one-more-discrete-log

Table 2. Comparisons amongst divisible on-line/off-line signature schemes. The word "stand." refers to operations or signature length of the underlying standard signature scheme. "Sig" in the assumption column means the security also depends on the security of the underlying standard signature scheme. Abbreviations used are: "sq." for squaring, and "mult." for multiplication.

Remark 4. Suppose g_i are in some group $\mathbb{G}$, e_i are all k-bit random values and t is small compared to k. By using a variant of the "square-and-multiply" method for exponentiation(Algorithm 14.88, [17]), computing $g_1^{e_1} g_2^{e_2} \dots g_t^{e_t}$ requires roughly k squarings and $(1 - \frac{1}{2^t})k$ multiplications in $\mathbb{G}$.

5 An Application to On-line/Off-line Threshold Signatures

Gennaro et al.[13] has proved that if a threshold signature scheme is simulatable, then its unforgeability can be reduced to the unforgeability of its underlying signature scheme. This provides a way to simplify the security proof of a threshold signature scheme. However, this result cannot be applied to on-line/off-line threshold signature schemes. Here we provide a similar result in Theorem 2 for

on-line/off-line threshold signature schemes. This theorem essentially states that a sufficient condition for the security reduction of an on-line/off-line threshold signature scheme is that the simulatability is "divisible".

Definition 7 (On-line/Off-line Threshold Signatures). *Let* $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$ *be a set of* n *players. An on-line/off-line threshold signature scheme* ($\mathcal{OTS}$) *for a divisible on-line/off-line signature scheme* $\mathcal{DOS} = (\mathsf{KeyGen}, \mathsf{Sign}^{\mathsf{off}}, \mathsf{Sign}^{\mathsf{on}}, \mathsf{Ver})$ *is a triple of algorithms* $(\mathsf{T\text{-}KeyGen}, \mathsf{T\text{-}Sign}, \mathsf{Ver})$.

- $(pk, sk, sk_1, \ldots, sk_n) \leftarrow \mathsf{T-KeyGen}(1^k)$. *The threshold key generation algorithm. It is a distributed PPT algorithm which on input a security parameter* $k \in \mathbb{N}$, *outputs i) a public/secret key pair* (pk, sk) *with the same distribution as* KeyGen, *and ii) the key shares* $sk_j (1 \leq j \leq n)$ *of* sk *where* sk *is known to nobody and* sk_j *is only known to* $P_j \in \mathcal{P}$.
- $\Sigma \leftarrow \mathsf{T\text{-}Sign}(sk_1, \ldots, sk_n, m)$. *The threshold signing algorithm. It is a distributed PPT algorithm which on input the secret key shares* $sk_j (1 \leq j \leq n)$ *and a message* m, *outputs a signature* Σ. *The signing algorithm consists of two sub-algorithms:*
  - $(\Sigma^{\mathrm{off}}, st_1, \ldots, st_n) \leftarrow \mathsf{T\text{-}Sign}^{\mathrm{off}}(sk_1, \ldots, sk_n)$. *The off-line phase of the algorithm* $\mathsf{T\text{-}Sign}$. *It is is a distributed PPT algorithm which on input the secret key shares* $sk_j (1 \leq j \leq n)$, *outputs an off-line signature token* Σ^{off} *and a state information* st_j *for each players* P_j.
  - $\Sigma^{\mathrm{on}} \leftarrow \mathsf{T\text{-}Sign}^{\mathrm{on}}(sk_1, \ldots, sk_n, st_1, \ldots, st_n, m)$. *The on-line phase of the algorithm* $\mathsf{T\text{-}Sign}$. *It is a distributed PPT algorithm which on input* sk_j, st_j *of* P_j *and a message* m, *outputs an on-line signature token* Σ^{on}.

  Finally, the signing algorithm returns $\Sigma = (\Sigma^{\mathrm{off}}, \Sigma^{\mathrm{on}})$ *as the signature for message* m. *The distribution of* $(\Sigma^{\mathrm{off}}, m, \Sigma^{\mathrm{on}})$ *is required to be the same as that of* $(\Sigma^{\mathrm{off}}_{\mathcal{DOS}}, m, \Sigma^{\mathrm{on}}_{\mathcal{DOS}})$ *where* $(\Sigma^{\mathrm{off}}_{\mathcal{DOS}}, St) \leftarrow \mathsf{Sign}^{\mathsf{off}}(sk)$ *and* $\Sigma^{\mathrm{on}}_{\mathcal{DOS}} \leftarrow \mathsf{Sign}^{\mathsf{on}}(sk, St, m)$.
- $0/1 \leftarrow \mathsf{Ver}(pk, m, \Sigma)$. *The verification algorithm. It is a PPT algorithm which on input the public key* pk, *a message* m *and a signature* Σ, *outputs 0 or 1 for* reject *or* accept *respectively. It is the same with the verification algorithm of* $\mathcal{DOS}$.

Robustness. Robustness of Scheme $\mathcal{OTS}$ means that the scheme will compute a correct output even in the presence of halting or malicious faults. Namely, even the inputs of some players to the algorithm $\mathsf{T\text{-}KeyGen}$ and $\mathsf{T\text{-}Sign}$ are absent or wrong, a robust scheme $\mathcal{OTS}$ can still be successfully finished.

Existential Unforgeability. Existential unforgeability for an on-line/off-line threshold scheme $\mathcal{OTS}$ (with a threshold t) is defined in the following game. This game is carried out between a challenger $\mathcal{S}$ and an adversary $\mathcal{A}$ who can corrupt up to t players. The adversary $\mathcal{A}$ is allowed to arouse the $\mathsf{T\text{-}Sign}$ algorithm polynomial times. We suppose $\mathcal{B}$ is the set of currently corrupted players. The attack game is as follows:

1. The challenger runs $\mathsf{T\text{-}KeyGen}$ on input 1^k to get $(pk, sk, sk_1, \ldots, sk_n)$. Let $\mathsf{View}_{\mathcal{A}}^{\text{T-KeyGen}}$ be $\mathcal{A}$'s view in this phase, which includes pk, $\{sk_i : P_i \in \mathcal{B}\}$ and other information.

2. On input $\mathsf{View}_{\mathcal{A}}^{\mathsf{T\text{-}KeyGen}}$, $\mathcal{A}$ is allowed to arouse $\mathsf{T\text{-}Sign}(sk_1, \ldots, sk_n, \cdot)$ (on behalf of the players in $\mathcal{B}$) polynomial times. The messages the adversary selected to query the signing oracle may depend on previous obtained signatures and views, i.e., the s-th message selected to query $\mathsf{T\text{-}Sign}$ can depend on all i-th signing view for $i = 1, \ldots, s-1$ and the s-th off-line signing view. At the end of this phase, $\mathcal{A}$ gets different messages signed.
3. $\mathcal{A}$ outputs a forgery (m, Σ).

The adversary wins the game if the message m has never been queried to the oracle $\mathsf{T\text{-}Sign}(sk_1, \ldots, sk_n, \cdot)$ and $\mathsf{Ver}(pk, m, \Sigma) = 1$ holds. Let $\mathrm{Adv}_{\mathcal{A},\mathcal{OTS}}$ be the *advantage* of the adversary $\mathcal{A}$ in breaking the signature scheme $\mathcal{OTS}$, i.e.,

$$\mathrm{Adv}_{\mathcal{A},\mathcal{OTS}} = \Pr[\mathsf{Ver}(pk, m, \Sigma) = 1 : (pk, sk, sk_1, \ldots, sk_n) \leftarrow \mathsf{T\text{-}KeyGen}(1^k); (m, \Sigma) \leftarrow \mathcal{A}^{\mathsf{T\text{-}Sign}(sk_1, \ldots, sk_n, \cdot)}]$$

where $\mathcal{A}$ has never requested m to the signing oracle and the probability is taken over the internal coin tosses of the algorithm $\mathsf{T\text{-}KeyGen}$ and $\mathcal{A}$.

Definition 8 (Unforgeability). *A scheme $\mathcal{OTS}$ is existentially unforgeable under adaptive chosen message attacks if for every PPT adversary $\mathcal{A}$, $\mathrm{Adv}_{\mathcal{A},\mathcal{OTS}}$ is negligible.*

Definition 9 (Simulatability). *An on-line/off-line threshold signature scheme $\mathcal{OTS} = (\mathsf{T\text{-}KeyGen}, \mathsf{T\text{-}Sign}, \mathsf{Ver})$ is simulatable if the following properties hold for any PPT adversary $\mathcal{A}$ against $\mathcal{OTS}$:*

1. *The algorithm $\mathsf{T\text{-}KeyGen}$ is simulatable. That is, there exists a PPT simulator $\mathsf{SIM}_{\mathsf{KeyGen}}$ that, on input the public key pk generated by an execution of $\mathsf{T\text{-}KeyGen}$, can simulate $\mathcal{A}$'s view $\mathsf{View}_{\mathcal{A}}^{\mathsf{T\text{-}KeyGen}}$.*
2. *The algorithm $\mathsf{T\text{-}Sign}$ is simulatable. Let the public key of $\mathcal{OTS}$ be pk and an execution of algorithm $\mathsf{T\text{-}Sign}$ be*

$$(\Sigma^{\mathrm{off}}, \Sigma^{\mathrm{on}}) \leftarrow \mathsf{T\text{-}Sign}(sk_1, \ldots, sk_n, m). \qquad (*)$$

The algorithm $\mathsf{T\text{-}Sign}$ is simulatable means there exists a PPT simulator $\mathsf{SIM}_{\mathsf{Sign}}$ which satisfies:

- *On input the public key pk, the off-line signature Σ^{off} and the transcript of the execution of $\mathsf{SIM}_{\mathsf{KeyGen}}$ (in particular the private information that is given to $\mathcal{A}$), $\mathsf{SIM}_{\mathsf{Sign}}$ can simulate $\mathcal{A}$'s off-line singing view $\mathsf{View}_{\mathcal{A}}^{\mathsf{Sign}^{\mathrm{off}}}$ in the process (*).*
- *After $\mathsf{SIM}_{\mathsf{Sign}}$ gets inputs m and Σ^{on}, it can also simulate $\mathcal{A}$'s on-line singing view $\mathsf{View}_{\mathcal{A}}^{\mathsf{Sign}^{\mathrm{on}}}$ in the process (*).*

Theorem 2. *An on-line/off-line threshold signature scheme $\mathcal{OTS}$ is existentially unforgeable under adaptive chosen message attacks, provided that it is simulatable and its underlying signature scheme is a divisible on-line/off-line signature scheme which is existentially unforgeable under adaptive chosen message attacks.*

Proof. (sketch). We prove this theorem by contradiction. Assume there exists an Algorithm $\mathcal{A}$ which breaks the unforgeability of $\mathcal{OTS}$ in polynomial time with non-negligible probability, then we construct an Algorithm $\mathcal{B}$ which breaks the unforgeability of its underlying divisible on-line/off-line signature scheme, say $\mathcal{DOS}$.

Let $\mathcal{DOS} = (\mathsf{KeyGen}, \mathsf{Sign}^{\mathsf{off}}, \mathsf{Sign}^{\mathsf{on}}, \mathsf{Ver})$. and $\mathcal{OTS} = (\mathsf{T\text{-}KeyGen}, \mathsf{T\text{-}Sign}, \mathsf{Ver})$. To break the unforgeability of $\mathcal{DOS}$, $\mathcal{B}$ works as follows.

Setup: Suppose the challenger runs KeyGen to get a public/private key pair (pk, sk) of $\mathcal{DOS}$ and pk is sent to $\mathcal{B}$. By the precondition, $\mathcal{OTS}$ is simulatable, so there exists a simulator $(\mathsf{SIM}_{\mathsf{KeyGen}}, \mathsf{SIM}_{\mathsf{Sign}})$ which can simulate $\mathcal{OTS}$'s key generation process and singing process. $\mathcal{B}$ runs $\mathsf{SIM}_{\mathsf{KeyGen}}$ with input pk to simulate $\mathcal{A}$'s view in Algorithm $\mathsf{T\text{-}KeyGen}$.

Simulating the signing oracle $\mathsf{T\text{-}Sign}$:

- When $\mathcal{A}$ arouses $\mathsf{T\text{-}Sign}^{\mathrm{off}}$, $\mathcal{B}$ first queries the oracle $\mathsf{Sign}^{\mathsf{off}}$, to get an off-line signature token Σ^{off}. Next, $\mathcal{B}$ runs $\mathsf{SIM}_{\mathsf{Sign}}$ with input Σ^{off} and the transcript of the execution of $\mathsf{SIM}_{\mathsf{KeyGen}}$ to simulate the off-line singing view of $\mathcal{A}$.
- When $\mathcal{A}$ arouses $\mathsf{T\text{-}Sign}^{\mathrm{on}}$ to sign a message m, $\mathcal{B}$ submits the query m to the oracle $\mathsf{Sign}^{\mathsf{on}}$, to get an on-line signature token Σ^{on}. Then, $\mathcal{B}$ continues to feed $\mathsf{SIM}_{\mathsf{Sign}}$ with input m and Σ^{on} to simulate $\mathcal{A}$'s off-line signing view.

Output: The simulated view of $\mathcal{A}$ is indistinguishable from the real one. Thus, $\mathcal{A}$ outputs a forgery (m, Σ) for $\mathcal{OTS}$ with non-negligible probability. $\mathcal{B}$ outputs (m, Σ) since it is also a valid forgery for $\mathcal{DOS}$ and thus the theorem is proved. ■

Theorem 2 provides an approach to construct on-line/off-line threshold signatures: given a divisible on-line/off-line signature scheme $\mathcal{DOS}$, we construct a scheme $\mathcal{TOS}$, which is a threshold version of $\mathcal{DOS}$. If $\mathcal{TOS}$ is simulatable in the sense of Definition 9, then $\mathcal{TOS}$ is a secure on-line/off-line threshold signature scheme.

References

[1] M. Bellare, C. Namprempre, D. Pointcheval, and M. Semanko. The One-More-RSA-Inversion problems and the security of Chaum's blind signature scheme. Report 2001/002, Cryptology ePrint Archive, May 2002.

[2] Mihir Bellare and Adriana Palacio. GQ and schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In *CRYPTO: Proceedings of Crypto*, 2002.

[3] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73. Springer, 2004.

[4] Emmanuel Bresson, Dario Catalano, and Rosario Gennaro. Improved on-line/off-line threshold signatures. In *Public Key Cryptography*, pages 217–232, 2007.

[5] Dario Catalano, Mario Di Raimondo, Dario Fiore, and Rosario Gennaro. Off-line/on-line signatures: Theoretical aspects and experimental results. In *Public Key Cryptography*, pages 101–120, 2008.

[6] Xiaofeng Chen, Fangguo Zhang, Willy Susilo, and Yi Mu. Efficient generic on-line/off-line signatures without key exposure. In *ACNS*, pages 18–30, 2007.

[7] Chris Crutchfield, David Molnar, David Turner, and David Wagner. Generic on-line/off-line threshold signatures. In *Public Key Cryptography*, pages 58–74, 2006.

[8] Y. Desmedt. Threshold cryptography. *European Transactions on Telecommunications*, 5(4):449–457, July 1994.

[9] Y. Desmedt. Some recent research aspects of threshold cryptography. *Lecture Notes in Computer Science*, 1396:158–173, 1998.

[10] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, IT-31(4):469–472, July 1985.

[11] Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital signatures. In G. Brassard, editor, *Proc. CRYPTO 89*, volume 435 of *Lecture Notes in Computer Science*, pages 263–277. Springer-Verlag, 1990.

[12] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew Michael Odlyzko, editor, *Advances in cryptology: CRYPTO '86: proceedings*, volume 263 of *Lecture Notes in Computer Science*, pages 181–187. Springer-Verlag, 1987.

[13] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Robust and efficient sharing of RSA functions. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 157–172. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1996.

[14] Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Robust threshold DSS signatures. In Ueli Maurer, editor, *Advances in Cryptology – EUROCRYPT '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 354–371. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1996.

[15] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988. Special issue on cryptography.

[16] Kaoru Kurosawa and Katja Schmidt-Samoa. New online/offline signature schemes without random oracles. In *Public Key Cryptography*, pages 330–346, 2006.

[17] A. J. (Alfred J.) Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. *Handbook of applied cryptography*. The CRC Press series on discrete mathematics and its applications. CRC Press, 2000 N.W. Corporate Blvd., Boca Raton, FL 33431-9868, USA, 1997.

[18] National Institute of Standards and Technology (NIST). The Digital Signature Standard. Federal Information Processing Standards Publication (FIPS PUB) 186, May 1994.

[19] Pascal Paillier and Damien Vergnaud. Discrete-log-based signatures may not be equivalent to discrete log. In *ASIACRYPT*, pages 1–20, 2005.

[20] Tal Rabin. A simplified approach to threshold and proactive RSA. In Hugo Krawczyk, editor, *Advances in Cryptology – CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 88–104. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 1998.

[21] Katja Schmidt-Samoa and Tsuyoshi Takagi. Paillier's cryptosystem modulo p^2q and its applications to trapdoor commitment schemes. In *Mycrypt*, pages 296–313, 2005.
[22] C. P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4:161–174, 1991.
[23] Adi Shamir and Yael Tauman. Improved online/offline signature schemes. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO '2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 355–367. International Association for Cryptologic Research, Springer-Verlag, Berlin Germany, 2001.
[24] Shidi Xu, Yi Mu, and Willy Susilo. Online/offline signatures and multisignatures for aodv and dsr routing security. In *ACISP*, pages 99–110, 2006.
[25] Ping Yu and Stephen R. Tate. An online/offline signature scheme based on the strong rsa assumption. In *AINA Workshops (1)*, pages 601–606, 2007.

In appendix, we give elliptic curve analogues of existing divisible on-line/off-line signature schemes in order to fairly compare them with our proposed scheme.

A. Crutchfield et al.'s Divisible On-line/Off-line Scheme CMTW-OS

The on-line/off-line signature scheme CMTW-OS is extracted from [7]. In [7], the authors construct an on-line/off-line threshold signature scheme which is a threshold version of this basic scheme. Let $\mathcal{S} = (\mathsf{G}, \mathsf{S}, \mathsf{V})$ be an ordinary signature scheme. The on-line/off-line signature scheme CMTW-OS $= (\mathsf{KeyGen}, \mathsf{Sign}^{\mathsf{off}}, \mathsf{Sign}^{\mathsf{on}}, \mathsf{Ver})$, where

- KeyGen.Choose an elliptic curve E over a finite field $\mathbb{F}$. Select a point in E with prime order p. Let $\mathbb{G}$ be a group generated by g. Choose $x \in_R \mathbb{Z}_p$, and let $h = g^x$. Run the key generation algorithm of $\mathcal{S}$ to obtain (pk, sk). The public key of $\mathcal{OS}$ is (E, p, g, h, pk), and the private key is (x, sk).
- $\mathsf{Sign}^{\mathsf{off}}$. (The i-th run). Choose $r, m \in_R \mathbb{Z}_p$, and compute $u = g^r h^m$. Use the signing algorithm of $\mathcal{S}$ to obtain $\sigma = \mathsf{S}_{sk}(u)$. Store the state information r, m. The off-line signature token is σ.
- $\mathsf{Sign}^{\mathsf{on}}$. (The i-th run, on a message $m' \in \mathbb{Z}_p$). Retrieve m, r from the memory. Compute $r' = r + (m - m')x \mod p$. The on-line signature token is r'.
- Ver. (On a message-signature pair (m', Σ) where $\Sigma = (\sigma, r')$). Verify that whether $\mathsf{V}_{pk}(g^{r'} h^{m'}, \sigma) = 1$.

Theorem 3. *The on-line/off-line signature scheme constructed above is divisible and existentially unforgeable under adaptive chosen message attacks, provided that the underlying signature scheme $\mathcal{S}$ is existentially unforgeable against generic chosen message attacks and the one-more-discrete-log assumption holds in $\mathbb{G}$.*

The proof of this theorem is omitted. Please refer to Theorem 2 of [7] for details. [6]

[6] The proof in [7] reduces the security of the on-line/off-line scheme CMTW-OS to the one-more-discrete-log assumption, or the collision resistance of a trapdoor hash

B. Bresson et al.'s Divisible On-line/Off-line Scheme BCG-OS

The on-line/off-line signature scheme BCG-OS is extracted from [4]. In [4], the authors construct an on-line/off-line threshold signature scheme which is a threshold version of this basic scheme. Let $\mathcal{S} = (\mathsf{G}, \mathsf{S}, \mathsf{V})$ be an ordinary signature scheme. The on-line/off-line signature scheme $\mathsf{BCG\text{-}OS} = (\mathsf{KeyGen}, \mathsf{Sign}^{\mathsf{off}}, \mathsf{Sign}^{\mathsf{on}}, \mathsf{Ver})$, where

- KeyGen.Choose an elliptic curve E over a finite field $\mathbb{F}$. Select a point in E with order power p. Let $\mathbb{G}$ be a group generated by g. Choose $x, y \in_R \mathbb{Z}_p$, and let $h_1 = g^x, h_2 = g^y$. Run the key generation algorithm of $\mathcal{S}$ to obtain (pk, sk). The public key is (E, p, g, h_1, h_2, pk), and the private key is (x, y, sk).
- $\mathsf{Sign}^{\mathsf{off}}$. (The i-th run). Choose $r, s, m \in_R \mathbb{Z}_p$, and compute $u = g^m h_1^r h_2^s$. Use the signing algorithm of $\mathcal{S}$ to obtain $\sigma = \mathsf{S}_{sk}(u)$. Store the state information r, s, m. The off-line signature token is σ.
- $\mathsf{Sign}^{\mathsf{on}}$. (The i-th run, on a message m'). Retrieve r, s, m from the memory. Choose $r' \in_R \mathbb{Z}_p$ and compute $s' = s + y^{-1}[(m - m') + (r - r')x] \mod p$. The on-line signature token is (r', s').
- Ver. (On a message-signature pair (m', Σ) where $\Sigma = (\sigma, r', s')$). Verify that whether $\mathsf{V}_{pk}(g^{m'} h_1^{r'} h_2^{s'}, \sigma) = 1$.

Remark 5. To reduce the on-line signing cost, we can move the selection of r' and computing $(r - r') \cdot x$ to the off-line phase. Thus, the on-line signing requires only 1 modular multiplication in $\mathbb{Z}_p$.

Theorem 4. *The on-line/off-line signature scheme constructed above is divisible and existentially unforgeable under adaptive chosen message attacks, provided that the underlying signature scheme $\mathcal{S}$ is existentially unforgeable against generic chosen message attacks and the discrete logarithm assumption holds in $\mathbb{G}$.*

The proof of this theorem is also omitted. Please refer to Theorem 1 of [4] for details.

C. Proving the Schnorr Signature Scheme[22] is Divisible

A variant of the Schnorr signature scheme can be naturally viewed as a divisible on-line/off-line signature scheme: $\mathsf{Schnorr\text{-}OS} = (\mathsf{KeyGen}, \mathsf{Sign}^{\mathsf{off}}, \mathsf{Sign}^{\mathsf{on}}, \mathsf{Ver})$.

- KeyGen.Choose an elliptic curve E over a finite field $\mathbb{F}$. Select a point in E with prime order p. Let $\mathbb{G}$ be a group generated by g. Choose $x \in_R \mathbb{Z}_p$, and let $h = g^x$. Let H be a hash function: $H : \{0,1\}^* \to \mathbb{Z}_p$. The public key is (E, p, g, h, H), and the private key is x.

function, or the unforgeability of $\mathcal{S}$. A little modification of this proof can simply reduce the security to the one-more-discrete-log assumption or the unforgeability of $\mathcal{S}$.

– $\mathsf{Sign}^{\mathsf{off}}$. (The i-th run). Choose $r \in_R \mathbb{Z}_p$, and compute $u = g^r$. Store the state information r. The off-line signature token is u.
– $\mathsf{Sign}^{\mathsf{on}}$. (The i-th run, on a message m). Retrieve r from the memory. Set $c = H(m\|u)$ and compute $s = r - cx \mod p$. The on-line signature token is s.
– Ver. (On a message-signature pair (Σ, m) where $\Sigma = (u, s)$. Verify that whether $g^s h^{H(m\|u)} = u$.

Remark 6. The signature token is defined as $(u, s) \in \mathbb{G} \times \mathbb{Z}_p$ instead of $(c, s) \in \mathbb{Z}_p \times \mathbb{Z}_p$. This is to decrease the on-line signature length because the value of c can't be computed in the off-line phase.

Lemma 2. *The divisible on-line/off-line signature scheme* Schnorr-OS *is existentially unforgeable against adaptive chosen message attacks in the random oracle model, provided that the one-more-discrete-log assumption holds in* $\mathbb{G}$.

Proof. Assume there exists an Algorithm $\mathcal{A}$ which $(t, q_{\mathrm{off}}, q_{\mathrm{on}}, q_{\mathrm{h}}, \epsilon)$ breaks the unforgeability of Schnorr-OS in the random-oracle based game defined in Section 3. Then we construct an Algorithm $\mathcal{B}$ which breaks the q_{on}-DL problem in polynomial time with probability $\epsilon' \geq (\frac{\epsilon}{q_{\mathrm{h}}} - \frac{1}{p})^2$.

Let g be a point in E with prime order p. Let $\mathbb{G}$ be a group generated by g. Algorithm $\mathcal{B}$ is given a q_{on}-DL problem instance $(\mathsf{descr}(\mathbb{G}), g, v_1, \ldots, v_{q_{\mathrm{on}}+1})$ where $\mathsf{descr}(\mathbb{G}) = (E, p)$. To find all the values of $\mathrm{dl}_g v_i$, Algorithm $\mathcal{B}$ works as follows:

Setup: Algorithm $\mathcal{B}$ sets $h = v_{q_{\mathrm{on}}+1}$ and gives to $\mathcal{A}$ the public key (E, p, g, h, H) where H is modeled as a random oracle defined in the following.

Simulating the hash oracle: Maintain a H-list of pairs, which is initialized to empty. Upon a query input $M \in \{0,1\}^*$, returns a random value $c \in \mathbb{Z}_p$ if M is not in the H-list. Then add (M, c) to the H-list.

Simulating the signing oracle (off-line): Upon the i-th query, return v_i as the off-line signature token.

Simulating the signing oracle (on-line): Upon the i-th query input m_i, Algorithm $\mathcal{B}$ makes query $m_i\|v_i$ to the hash oracle to obtain an answer, say c_i. Next, Algorithm $\mathcal{B}$ makes query $v_i h^{-c_i}$ to the oracle $\mathsf{DL}_g(\cdot)$ to obtain $w_i = \mathrm{dl}_g(v_i h^{-c_i})$ and returns w_i as the answer for the query m_i. $\mathcal{B}$ makes at most q_{on} queries to the oracle $\mathsf{DL}_g(\cdot)$ since $\mathcal{A}$ makes at most q_{on} on-line signing queries.

Rewinding: At the beginning of the game, Algorithm $\mathcal{B}$ randomly selects a value $d \in q_{\mathrm{h}}$. Upon the d-th hash query, the simulated hash oracle returns a random value $c \in \mathbb{Z}_p$ as usual. If $\mathcal{A}$ successfully outputs a valid signature forgery on a message, Algorithm $\mathcal{B}$ then resets $\mathcal{A}$ to the step where $\mathcal{A}$ has just sent the d-th hash query. This time, the simulated hash oracle again randomly selects a value, say c', and returns it as the answer. Next, $\mathcal{B}$ continues the game which runs the second instance of $\mathcal{A}$, which has the same inputs and internal coins with the first instance.

Output: Let the d-th hash query be $m\|t \in \{0,1\}^* \times \mathbb{Z}_p$. Algorithm $\mathcal{B}$ successfully ends the game if the following events occur:

1) The first instance of $\mathcal{A}$ successfully outputs a valid signature forgery (t, s_1) for the message m where s_1 is some value in $\mathbb{Z}_p$, and
2) The second instance of $\mathcal{A}$ successfully outputs a valid signature forgery (t, s_2) for the message m where s_2 is some value in $\mathbb{Z}_p$, and
3) $c \neq c'$.

If the above events occur, we can conclude that $g^{s_1}h^c = g^{s_2}h^{c'} = t$ and $c \neq c'$. Thus, $\mathcal{B}$ can output the solution of the q_{on}-DL problem as follows:
$\text{dl}_g(v_{q_{\text{on}}+1}) = \text{dl}_g(h) = (s_1 - s_2)(c' - c)^{-1} \mod p$.
$\text{dl}_g(v_i) = w_i + c_i \text{dl}_g(v_{q_{\text{on}}+1}) \mod p$ for i from 1 to q_{on}.

Let's estimate the probability that $\mathcal{B}$ successfully ends the game. By a standard argument similar to the reset lemma of [2], we get that $\mathcal{B}$ succeeds with probability at least $(\frac{\epsilon}{q_{\text{h}}} - \frac{1}{p})^2$. Thus, $\mathcal{B}$ successfully breaks q_{on}-DL problem in polynomial time with non-negligible probability. This contradicts the one-more-discrete-log assumption and thus the theorem is proved. ■