Digital Signatures from Challenge-Divided Σ-Protocols

Andrew C. Yao * Yunlei Zhao†

Abstract

Digital signature is one of the basic primitives in cryptography. A common paradigm of obtaining signatures, known as the Fiat-Shamir (FS) paradigm, is to collapse any Σ-protocol (which is 3-round public-coin honest-verifier zero-knowledge) into a non-interactive scheme with hash functions that are modeled to be random oracles (RO). The Digital Signature Standard (DSS) and Schnorr's signature schemes are two salient examples following the FS-paradigm.

In this work, we present a modified Fiat-Shamir paradigm, named *challenge-divided Fiat-Shamir paradigm*, which is applicable to a variant of Σ-protocol with *divided* random challenges. This new paradigm yields a new family of (*online/offline efficient*) digital signatures from challenge-divided Σ-protocols, including in particular a variant of Schnorr's signature scheme called *challenge-divided Schnorr signature*. We then present a formal analysis of the challenge-divided Schnorr signature in the random oracle model. Finally, we give comparisons between the challenge-divided Schnorr signature and DSS and Schnorr's signature, showing that the newly developed challenge-divided Schnorr signature can enjoy better (online/offline) efficiency (besides provable security in the random oracle model).

Of independent interest is a new forking lemma, referred to as *divided* forking lemma, for dealing with multiple ordered rewinding points in the RO model, which is of independent interest and can be applied to analyzing other cryptographic schemes in the RO model.

1 Introduction

Digital signature is fundamental to modern cryptography. A common paradigm of obtaining signatures, known as the Fiat-Shamir (FS) paradigm [5], is to collapse any Σ-protocol [3] into a non-interactive scheme with hash functions that are modeled to be random oracles (RO) [1]. Roughly speaking, denote by a, e, z the first, the second and the third message of a Σ-protocol respectively, where a and z are from the prover and e is a random challenge from the verifier. Given a message $m \in \{0,1\}^*$ to be signed, the FS-paradigm computes and outputs (e, z) as the signature, where $e = h(a, m)$ and h is a hash function that is modeled to be a random oracle in the security analysis. To improve the online/offline efficiency of digital signatures instantiated via the FS-paradigm, the signer can pre-compute and store a list of values a's.

Two salient examples following the FS-paradigm is the Digital Signature Standard (DSS) [6] and Schnorr's signature [16]. The formal analysis of digital signatures via the FS-paradigm was conducted in [15], where the core of the analysis is a forking lemma. The forking lemma given in [15] is for analyzing cryptographic schemes with a single rewinding point in the random oracle model. But, in some recent applications, security analysis in the random oracle model may need to deal with multiple ordered rewinding points.

The notion of *online/offline signature* is introduced in [4]. The idea is to perform signature generation into two phases: the offline phase and the online phase. On-line/offline signature schemes are useful, since in many applications the signer (e.g., a smart-card) has a very limited response time once the message is presented (but it can carry out costly computations between consecutive signing requests). The online phase is typically very fast, and hence can be executed even on a weak processor. On-line/offline

*Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China. andrewcyao@tsinghua.edu.cn

†Software School, Fudan University, Shanghai 200433, China. ylzhao@fudan.edu.cn

signature schemes are particularly remarkable in smart-card based applications [17]: the offline phase can be implemented either during the card manufacturing process or as a background computation whenever the card is connected to power. Some general transformations from any signature scheme to secure online/offline signature scheme are know (e.g., [4, 17]), but typically are not as efficient (in terms of both computational complexity and space complexity of the signer) as the signature schemes resultant directly via the Fiat-Shamir paradigm.

In this work, we present a modified Fiat-Shamir paradigm, named *challenge-divided Fiat-Shamir paradigm*, which is applicable to a variant of Σ-protocol with *divided* random challenges (referred to as *challenge-divided* Σ-protocol). This yields a new family of *online/offline efficient* digital signatures from challenge-divided Σ-protocols, where the signer particularly does not need to store the values a's. We present a formal analysis of the challenge-divided Schnorr signature in the random oracle model. Along the way, we prove a new forking lemma, referred to as *divided* forking lemma, for dealing with multiple ordered rewinding points in the random oracle model, which is of independent interest and can be applied to analyzing other cryptographic schemes in the random oracle model. Finally, we give comparisons between the challenge-divided Schnorr signature and DSS and Schnorr's signature, showing that the newly developed challenge-divided Schnorr signature can enjoy better (online/offline) efficiency (besides provable security in the random oracle model in view that the provable security of DSS is still unknown).

2 Preliminaries

If A is a probabilistic algorithm, then $A(x_1, x_2, \cdots; r)$ is the result of running A on inputs $x_1, x_2, \cdots$ and coins r. We let $y \leftarrow A(x_1, x_2, \cdots; r)$ denote the experiment of picking r at random and letting y be $A(x_1, x_2, \cdots; r)$. If S is a finite set then $x \leftarrow S$, sometimes also written as $x \in_{\mathrm{R}} S$, is the operation of picking an element uniformly from S. If α is neither an algorithm nor a set then $x \leftarrow \alpha$ is a simple assignment statement. A function $f(\lambda)$ is *negligible* if for every $c > 0$ there exists a λ_c such that $f(\lambda) < \frac{1}{\lambda^c}$ for all $\lambda > \lambda_c$.

Let G' be a finite Abelian group of order N, G be a subgroup of prime order q in G'. Denote by g a generator of G, by 1_G the identity element, by $G \setminus 1_G = G - \{1_G\}$ the set of elements of G except 1_G. Throughout this paper, unless explicitly specified, for presentation simplicity we assume G' is Z_p^* (of order $N = p - 1$) where p is a large prime, and use multiplicative notation to describe the group operation in G'. (When G' is defined w.r.t. elliptic curves over finite fields, usually addition notation is used for the group operation in G'.) On input of the form (p, q, g, X) such that $X = g^x \ mod \ p$, where p, q are primes, g is an element in Z_p^* of order q and x is taken uniformly at random from Z_q^*, the discrete logarithm problem (DLP) is to compute x, and the DLP assumption says that no probabilistic polynomial-time (PPT) algorithm can solve the DLP problem with non-negligible probability for all sufficiently large q. The DLP problem and DLP assumption can also be defined on elliptic curves over finite fields.

2.1 Σ-Protocols

Definition 2.1 (Σ-protocol [3]) *A three-round public-coin protocol $\langle P, V \rangle$ is said to be a Σ-protocol for an $\mathcal{NP}$-relation $\mathcal{R}$ if the following hold:*

- *Completeness. If P, V follow the protocol, the verifier always accepts.*
- *Special soundness. From any common input U of length n and any pair of accepting conversations on input U, (a, e, z) and (a, e', z') where $e \neq e'$, one can efficiently compute w such that $(U, w) \in \mathcal{R}$ with overwhelming probability. Here a, e, z stand for the first, the second and the third message respectively and e is assumed to be a string of length l (that is polynomially related to n) selected uniformly at random in $\{0,1\}^l$.*

- *Perfect/statistical SHVZK (special honest verifier zero-knowledge). There exists a probabilistic polynomial-time simulator S, which on input U (where there exists an $\mathcal{NP}$-witness w such that $(U, w) \in \mathcal{R}$) and a random challenge string $\hat{e}$, outputs an accepting conversation of the form $(\hat{a}, \hat{e}, \hat{z})$, with the same probability distribution as that of the real conversation (a, e, z) between the honest $P(w)$, V on input U.*

The first Σ-protocol (for an $\mathcal{NP}$-language) in the literature can be traced back to the honest verifier zero-knowledge (HVZK) protocol for Graph Isomorphism [7] (but the name of Σ-protocol is adopted much later in [3]), and a large number of Σ-protocols for various languages have been developed now. Σ-protocols have been proved to be a very powerful cryptographic tool, and are widely used in numerous important cryptographic applications. Below, we briefly recall the Σ-protocol examples for DLP and RSA.

Σ-Protocol for DLP [16]. The following is a Σ-protocol $\langle P, V \rangle$ proposed by Schnorr [16] for proving the knowledge of discrete logarithm, w, for a common input of the form (p, q, g, U) such that $U = g^w \ mod \ p$, where p, q are primes and g is an element in Z_p^* of order q. Normally, the length of q, denoted $|q|$, is served as the security parameter.

- P chooses r at random in Z_q and sends $a = g^r \ mod \ p$ to V.
- V chooses a challenge e at random in Z_{2^l} and sends it to P. Here, l is fixed such that $2^l < q$.
- P sends $z = r + ew \ mod \ q$ to V, who checks that $g^z = aU^e \ mod \ p$, that p, q are prime and that g, h are of order q, and accepts iff this is the case.

Σ-Protocol for RSA [9]. Let n be an RSA modulus and q be a prime. Assume we are given some element $y \in Z_n^*$, and P knows an element w such that $w^q = y \mod n$. The following protocol is a Σ-protocol for proving the knowledge of q-th roots modulo n.

- P chooses r at random in Z_n^* and sends $a = r^q \mod n$ to V.
- V chooses a challenge e at random in Z_{2^l} and sends it to P. Here, l is fixed such that $2^l < q$.
- P sends $z = rw^e \mod n$ to V, who checks that $z^q = ay^e \mod n$, that q is a prime, that $gcd(a, n) = gcd(y, n) = 1$, and accepts iff this is the case.

2.2 The Fiat-Shamir Paradigm and Its Provable Security in the RO Model

Given *any* Σ-protocol (a, e, z) on common input U (which will be viewed as the signing public-key), the Fiat-Shamir paradigm collapses the Σ-protocol into a signature scheme as follows: $(a, e = h(a, m), z)$, where m is the message to be signed and h is a hash function. Note that, in actual signature schemes with the Fiat-Shamir paradigm, the generated signature only consists of (e, z) as the value a can be computed from (e, z). The provable security of the general Fiat-Shamir paradigm is shown by Pointcheval and Stern [15] in the random oracle model (assuming h to be an idealized random function). The core of the security arguments of Pointcheval and Stern [15] is a forking lemma.

Schnorr's signature scheme. The signature scheme obtained by applying the Fiat-Shamir paradigm on the above Schnorr's Σ-protocol for DLP is referred to as Schnorr's signature scheme. Note that, for Schnorr's signature scheme, the signer can pre-compute and store a list of values $(a = g^r, r)$. Then, to sign a message m, it simply computes $e = h(a, m)$ and z. That is, the signer only needs to perform $z = r + h(m, a)w$ online, where $a = g^r$ and r are offline pre-computed and stored.

2.3 The Digital Signature Standard (DSS)

The general structure of DSS is as follows:

- Public-key: $U = g^w \in G'$, where $w \in Z_q^*$. Typically, w is a 160-bit prime.

- Secret-key: w.
- Signature generation: Let $m \in \{0,1\}^*$ be the message to be signed.
  1. Compute $a = g^r \mod p$, where r is taken randomly from Z_q. Compute $d = f(a)$, where $f : G' \to Z_q^*$ is a conversion function.[1]
  2. Compute s from the equation $h(m) = sr - dw \mod q$, as follows:
     - Compute $\hat{r} = r^{-1}$.
     - Compute $s = (h(m) + dw)\hat{r}$, or $s = h(m)\hat{r} + dw\hat{r}$ with offline pre-computed $dw\hat{r}$, where h is a hash function.
  3. Output (d, s) as the signature.
- Signature verification: Given $(e = h(m), d, s)$ where $d, s \in Z_q^*$, the verifier verifies the signature as follows:
  - Compute $\hat{s} = s^{-1}$.
  - Verify $f(g^{e\hat{s}}U^{d\hat{s}}) = d$, where $e = h(m)$.

Recall that in the DSS scheme, the signature is generated as: $(d, s = er^{-1} + dwr^{-1})$, where $e = h(m)$, $d = f(a)$ and $a = g^r$, w is the secret-key. In general, the conversion $f : G' \to Z_q^*$ also can be viewed as RO. Observe that the value m (i.e., the message to be signed) and the value $a = g^r$ are not put into the input of a single RO in the DSS scheme, contrary to signature schemes via the Fiat-Shamir scheme where (m, a) is put into the single RO h. The separation of m and a in the inputs of random oracles and the way of signature generation of DSS bring the following advantages to DSS over Schnorr's signature scheme.

Specifically, the signer can pre-compute a list of values a's (just as in signature schemes via the Fiat-Shamir paradigm), but contrary to signature schemes via the Fiat-Shamir paradigm, the signer of DSS does not need to store these pre-computed values. Specifically, for each pre-computed value $a = g^r$, the DSS signer can offline compute $d = f(a)$, r^{-1} and dwr^{-1}, and only stores (d, r^{-1}, dwr^{-1}) (note that it is unreasonable to assume the message to be signed is always known beforehand). *Actually, for smart-card based applications, the values* (d, r^{-1}, dwr^{-1})*'s can be stored during the card manufacturing process.* Note that $d, r^{-1}, dwr^{-1} \in Z_q$, while $a \in Z_p^*$. As p is typically of 1024 bits while q is of 160 bits, and assuming the signer pre-computes k values of a's, then in comparison with Schnorr's signature scheme the space complexity (of storing pre-computed values) is reduced from $(|p| + |q|)k$ to $3|q|k$.

DSS vs. Schnorr's Signature The DSS scheme [6] is a variant of Schnorr's signature [16] via the Fiat-Shamir paradigm. The DSS scheme was proposed (1) to add online/offline efficiency by avoiding storing the value a's (and some more useful properties, see e.g., [14]), and (2) to bypass the patent issues related to Schnorr's signature scheme, but at the price of losing provable security.

In a sense, the challenge-divided Schnorr signature (instantiated by our challenge-divided Fiat-Shamir paradigm to be described in Section 3) can also be viewed as another way to avoid storing the values a's, but enjoys much further better (online/offline) efficiency than DSS and enjoys provable security in the RO model as well.

2.4 Existential Unforgeability Under Chosen Message Attack for Digital Signature Scheme

Formally, a digital signature scheme consists of three algorithms $(KeyGen, Sign, Verify)$. $KeyGen$ is the key generation algorithm, which on input a security parameter n outputs a pair of public-key and secret-key (PK, SK). $Sign$ is the signing algorithm, which takes the secret-key SK and a message $m \in \{0,1\}^*$ and outputs a signature σ. $Verify$ is the signature verification algorithm, which on input

[1] Typically, f can simply be the "mod q" operation for implementations over Z_p^*, or the operation of just taking the x-coordinate for implementations over elliptic curves over finite fields.

the public-key PK, a message m and a proposed signature σ outputs 1 (indicating verification success) or 0 (indicating verification failure).

Existential unforgeability under chosen message attacks for a signature scheme $(KeyGen, Sign, Verify)$ [8] is defined using the following game between a challenger and an adversary $\mathcal{A}$.

Setup. On a security parameter n, the challenger runs $KeyGen(1^n)$ to obtain a pair of public-key and secret-key (PK, SK). The public-key PK is given to adversary $\mathcal{A}$ (while the secret-key keeps private).

Adaptive queries. $\mathcal{A}$ *adaptively* requests signatures w.r.t. PK on at most R messages of its choice $M_1, \cdots, M_R \in \{0,1\}^*$, where R is a polynomial in n. The challenger responds to each query with a signature $\sigma_i = Sign(SK, M_i)$.

Output. Finally, $\mathcal{A}$ outputs a pair (M, σ), and wins the game if (1) $M \notin \{M_1, \cdots, M_R\}$, and (2) $Verify(PK, M, \sigma) = 1$.

We define $AdvSig_{\mathcal{A}}$ to be the probability that $\mathcal{A}$ wins in the above game, taken over the coin tosses of $KeyGen$ and of $\mathcal{A}$. Then, we say the signature scheme is *existentially unforgeable under adaptive chosen-message attacks* if for all sufficiently large n and any probabilistic polynomial-time adversary $\mathcal{A}$, it holds that $AdvSig_{\mathcal{A}}$ is negligible (in n).

3 Challenge-Divided Σ-Protocols, and Challenge-Divided Fiat-Shamir Paradigm

In this section, we show a modified Fiat-Shamir paradigm, named *challenge-divided Fiat-Shamir paradigm*, that is applicable to a variant of Σ-protocol with *divided* random challenges (that is referred to as *challenge-divided* Σ-protocol). Below, we first describe the challenge-divided Σ-protocols for DLP and RSA.

Challenge-divided Σ-Protocol for DLP. The common input is the same as that of Schnorr's protocol for DLP: (p, q, g, U) such that $U = g^w \ mod \ p$.

- P chooses r at random in Z_q and sends $a = g^r \ mod \ p$ to V.
- V chooses a *pair* of challenges d, e at random in $Z_{2^l} \times Z_{2^l}$ and sends (d, e) to P. Here, l is fixed such that $2^l < q$.
- P sends $z = er + dw \ mod \ q$ (resp., $z = dr + ew$) to V, who checks that $g^z = a^e U^d \ mod \ p$ (resp., $g^z = a^d U^e \ mod \ p$), that p, q are prime and that g, h are of order q, and accepts iff this is the case.

Challenge-divided Σ-Protocol for RSA. Let n be an RSA modulus and q be a prime. The common input is (n, q, y), and the private input is w such that $y = w^q \mod n$.

- P chooses r at random in Z_n^* and sends $a = r^q \mod n$ to V.
- V chooses a *pair* of challenges d, e at random in $Z_{2^l} \times Z_{2^l}$ and sends (d, e) to P. Here, l is fixed such that $2^l < q$.
- P sends $z = r^d w^e \mod n$ (resp., $z = r^e w^d \mod n$) to V, who checks that $z^q = a^d y^e \mod n$ (resp., $z^q = a^d y^e \mod n$), that q is a prime, that $gcd(a, n) = gcd(y, n) = 1$, and accepts iff this is the case.

The challenge-divided Fiat-Shamir paradigm for challenge-divided Σ-protocols. Let F be a one-way function (OWF) admitting challenge-divided Σ-protocols, i.e., the range of the OWF has a challenge-divided Σ-protocol for proving the knowledge of the corresponding preimage w.r.t. the $\mathcal{NP}$-relation $\{(U, w) | U = F(w)\}$. Let the random challenge be of length Len. Denote by d, e the (divided)

random challenges, and let $U = F(w)$ be signer's public-key and w the secret-key. To sign a message m, the signer computes a, $d = \tilde{f}(a)$, $e = \tilde{h}(m)$, and z, and then outputs (d, z) as the signature on m, where $\tilde{h}$ and $\tilde{f}$ are conversion functions from $\{0,1\}^*$ to $\{0,1\}^{Len}$. In security analysis in the RO model, we assume both $\tilde{h}$ and $\tilde{f}$ are random oracles.

4 *Challenge-Divided* Schnorr Signature Scheme, and Its Provable Security in the RO Model

With the above challenge-divided Schnorr's Σ-protocol for DLP as an illustrative instance, the transformed signature via the above challenge-divided Fiat-Shamir paradigm is called *challenge-divided* Schnorr signature. Note that for signatures from the above challenge-divided Schnorr's Σ-protocol for DLP, we have that $\tilde{f} = f$ and $\tilde{h} = h$ are conversion functions from $\{0,1\}^*$ to Z_q^*.[2] In the following, we directly describe the online/offline version of challenge-divided Schnorr signature.

- Public-key: $U = g^{-w} \in G'$, where $w \in Z_q^*$.
- Secret-key: w.
- Message to be signed: $m \in \{0,1\}^*$.
- Offline pre-computation: the signer pre-computes and stores (r, d, dw) (resp., (d, rd)), where r is taken randomly by the signer from Z_q^*, $a = g^r$, $d = f(a)$. The signature verifier can pre-compute $e = h(m)$ and $\hat{e} = e^{-1}$, in case it knows m before receiving the signature.
- Online signature generation: After receiving the message m to be signed, the signer computes $e = h(m)$, retrieves the pre-stored value (r, d, dw) (resp., (d, dr)), and computes $z = er + dw$ (resp., $z = dr + ew$). The signer outputs (d, z) as the signature on m.
- Signature verification: given a signature $(e = h(m), d, z)$ where $d, z \in Z_q^*$, check that $d, z \in Z_q^*$ and $f(g^{z\hat{e}}U^{d\hat{e}}) = d$ (resp., $f(g^{z\hat{d}}U^{e\hat{d}}) = d$), where $\hat{e} = e^{-1}$ (resp., $\hat{d} = d^{-1}$). Note that $\hat{e} = e^{-1}$ can be offline pre-computed by the verifier, in case it knows the message m before receiving the signature.

Theorem 4.1 *Assuming $h, f : \{0,1\}^* \to \{0,1\}^l/\{0\} \subseteq Z_q^*$ are random oracles where l is the security parameter, the challenge-divided Schnorr signature scheme is existentially unforgeable against adaptive chosen message attacks under the DLP assumption.*

Proof. We mainly provide the proof for challenge-divided Schnorr with $z = er + dw$, the proof for the variant of $z = dr + ew$ is similar.

Given a polynomial-time and successful forger $\mathcal{F}$, i.e., $\mathcal{F}$ successfully outputs (after polynomially many adaptively chosen queries to the signing oracle and random oracles), with non-negligible probability in polynomial-time, a valid signature on a new message that is different from those queried to the signing oracle, we build an efficient solver $\mathcal{C}$ for the DLP problem, namely, $\mathcal{C}$ gets as input a random element $U = g^{-w}$ in G and outputs the corresponding discrete logarithm w also with non-negligible probability. *For presentation simplicity, we assume the random oracles h, f are identical, namely we use the unique RO h to handle all RO queries $e = h(m)$ and $d = h(a)$.* The algorithm $\mathcal{C}$ is presented in Figure 1.

For the description of $\mathcal{C}$ in Figure 1, suppose $\mathcal{F}$ makes Q RO queries and R signing oracle queries (where Q and R are some polynomials in the security parameter l), we have the following proposition:

Proposition 4.1 *With probability at most $(RQ + R^2/2)/(q-1)$, $\mathcal{C}$ fails in one of Step S3 of signature simulations. $\mathcal{C}$ fails at Step F3 with probability at most $\frac{Q+2}{2^l-1}$.*

[2] As with DSS, in practice, f can simply be the "mod q" operation for implementations over Z_p^*, or the operation of just taking the x-coordinate for implementations over elliptic curves over finite fields.

Building the DLP solver $\mathcal{C}$ from the challenge-divided Schnorr forger $\mathcal{F}$

Setup: The input to $\mathcal{C}$ is a random element $U = g^{-w}$ in G, and its goal is to compute w. To this end, $\mathcal{C}$ provides $\mathcal{F}$ with a random tape, and runs the forger $\mathcal{F}$ as the challenge-divided Schnorr signer of public-key U.
RO queries: $\mathcal{C}$ provides random answers to queries to the random oracle h, under the limitation that if the same h query is presented more than once, $\mathcal{C}$ answers it with the same response as in the first time.
Signature query simulation: Each time $\mathcal{F}$ queries the signing oracle for a challenge-divided Schnorr signature on message m_i, $1 \leq i \leq R$, chosen by $\mathcal{F}$ adaptively, where m_i denotes the message in the i-th query, $\mathcal{C}$ answers the query as follows (note that $\mathcal{C}$ does not know the secret-key w corresponding to the public-key $U = g^w$):

S1. Chooses $z_i \in_{\mathrm{R}} Z_q^*$ and $d_i \in_{\mathrm{R}} \{0,1\}^l \subseteq Z_q^*$ where l is the output length of the RO h. If $h(m)$ has been defined by previous query to h, then sets $e_i = h(m)$, otherwise chooses $e_i \in_{\mathrm{R}} \{0,1\}^l \setminus \{0\}$ and defines $h(m) = e_i$.

S2. Computes $a_i = g^{z_i e_i^{-1}} U^{d_i e_i^{-1}}$.

S3. If $h(a_i)$ has been previously defined, $\mathcal{C}$ aborts its run and outputs "fail". Otherwise, sets $h(a_i) = d_i$. Recall that, for presentation simplicity, we have assumed $f = h$.

S4. $\mathcal{C}$ responds to $\mathcal{F}$'s signing query m_i with the simulated signature (d_i, z_i).

When $\mathcal{F}$ halts, $\mathcal{C}$ checks whether the following conditions hold:

F1. $\mathcal{F}$ outputs (m, d, z) such that (d, z) is a valid signature on m. That is, d, z are in Z_q^*, $e = h(m)$ $a = g^{ze^{-1}} U^{de^{-1}}$, and $d = h(a)$.

F2. m was not queried by $\mathcal{F}$ to the signing oracle previously, i.e., $m \neq m_i$ for all $i, 1 \leq i \leq R$.

F3. The values $h(m)$ and $h(a)$ were queried from the RO h.

If these three conditions hold, $\mathcal{C}$ proceeds to the "repeat experiments" below; in all other cases $\mathcal{C}$ halts and outputs "fail".
The repeat experiments. $\mathcal{C}$ runs $\mathcal{F}$ again for a second time, under the same public-key U and using the same coins for $\mathcal{F}$. There are two cases according to the order of the RO queries of $h(m)$ and $h(a)$:

C1. $h(m)$ posterior to $h(a)$: $\mathcal{C}$ rewinds $\mathcal{F}$ to the point of making the RO query $h(m)$, responds back a new independent value $e' \in_{\mathrm{R}} \{0,1\}^l \setminus \{0\}$. All subsequent actions of $\mathcal{C}$ (including random answers to subsequent RO queries) are independent of the first run. If in this repeated run $\mathcal{F}$ outputs a valid signature (d, z') for the message m, i.e., $e' = h(m)$, $d = h(a)$ and $a = g^{z'e'^{-1}} U^{de'^{-1}}$, $\mathcal{C}$ computes $w = (z'e'^{-1} - ze^{-1})/(de'^{-1} - de^{-1}) \mod q$.

C2. $h(a)$ posterior to $h(m)$: $\mathcal{C}$ rewinds $\mathcal{F}$ to the point of making the RO query $h(a)$, responds back a new independent value $d' \in_{\mathrm{R}} \{0,1\}^l \setminus \{0\}$. All subsequent actions of $\mathcal{C}$ (including random answers to subsequent RO queries) are independent of the first run. If in this repeated run $\mathcal{F}$ outputs a valid signature (d', z') for the message m, i.e., $e = h(m)$, $d' = h(a)$ and $a = g^{z'e^{-1}} U^{d'e^{-1}}$, $\mathcal{C}$ computes $w = (z' - z)/(d' - d) \mod q$.

Figure 1: Reduction from DLP to challenge-divided Schnorr signature forgeries

Proof (of Proposition of 4.1). It is easy to check that suppose $\mathcal{C}$ never fails at Step S3, the signature simulations by $\mathcal{C}$ are of identical distribution with that of real signatures by using the secret-key w.

Next, we limit the upper-bound of Step S3 failure. Note that for each a_i generated by $\mathcal{C}$ at Step S2, it is distributed uniformly in $G \setminus 1_G$. In the RO model, there are two cases for $\mathcal{C}$ fails at Step S3:

- For *some* i, $1 \leq i \leq R$, $\mathcal{F}$ ever successfully guessed the value a_i in one of its Q random oracle queries. Thus, the probability that $\mathcal{C}$ fails in Case 1 is at most $RQ/(q-1)$.
- For some i, $1 \leq i \leq R$, the value a_i has ever been generated in dealing with the j-th signing oracle query, $j < i$. The probability that $\mathcal{C}$ fails in Case 2 is at most $C_R^2/(q-1) \leq (R^2/2)/(q-1)$, where C_R^2 is the combination number of selecting two elements from a set of R elements.

Finally, it is easy to check that $\mathcal{C}$ fails at Step F3 with probability at most $\frac{Q+2}{2^l-1}$. To see this, we first consider the value d. There are two cases:

Case-1: The value d is undefined (specifically, d was not output by the RO h in the simulation, i.e., there exists no a record of the form $(\cdot, d)$ in the simulation of the RO h). This case implies that $\mathcal{F}$ simply guesses the value d, which can succeed with probability $\frac{1}{2^l-1}$.

Case-2: The value d is defined. We further consider two cases:

Case-2.1: $\mathcal{F}$ did not make the RO query $h(m)$. In this case, the value a is also undefined, and for any $\alpha \in G \setminus 1_G$, $\Pr[a = \alpha] \leq \frac{1}{2^l-1}$. Thus, $\mathcal{F}$ can succeed with probability at most $\frac{Q}{2^l-1}$ in this case.

Case-2.2: $\mathcal{F}$ made the RO query $h(m) = e$, which then determines the value a, but $\mathcal{F}$ did not make the RO query $h(a)$. In this case, $\mathcal{F}$ can succeed (i.e., $\Pr[h(a) = d]$) with probability at most $\frac{1}{2^l-1}$.

Putting all together, we have that $\mathcal{C}$ can abort at Step F.3 with probability at most $\frac{Q+2}{2^l-1}$. □

Thus, suppose the forger $\mathcal{F}$ succeeds (i.e., outputs a valid signature (d, z) for a new message m different from those queried) with non-negligible probability in its real attack against the signer of public-key U, $\mathcal{F}$ succeeds in the first run of $\mathcal{C}$ in Figure 1 also with non-negligible probability (up to a gap at most $(QR + R^2/2)/(q-1)$). Then, with non-negligible probability (with a gap at most $(QR + R^2/2)/(q-1) + \frac{Q+2}{2^l-1}$ to the success probability of $\mathcal{F}$ in its real attack), $\mathcal{C}$ does the repeated second run.

For presentation simplicity, we write the signature of challenge-divided Schnorr on a message m as $(m, e = h(m), a, d = h(a), z)$. Note that given a pair of different signatures on the same m (and a): $\{(m, e, a, d, z), (m, e', a, d, z')\}$ that corresponds to Case C1 in Figure 1, or, $\{(m, e, a, d, z), (m, e, a, d', z')\}$ that corresponds to Case C2 in Figure 1, the value w computed by $\mathcal{C}$ is correct. Thus, to finish the theorem, what left is to show that conditioned $\mathcal{F}$ succeeds in outputting the valid (m, e, a, d, z) in the first run of $\mathcal{C}$, with non-negligible probability $\mathcal{F}$ will also succeed in Case C1 or Case C2 of the repeated second run. We note that this can be shown by an *extended* version of the Pointcheval-Stern forking lemma [15] (that was originally developed to argue the security of digital signature schemes via the Fiat-Shamir paradigm). For completeness, we reproduce the forking lemma tailored for signature schemes via the challenge-divided Fiat-Shamir paradigm, referred to as *divided* forking lemma.

Suppose $\mathcal{F}$ produces, with probability ε', a valid signature (m, e, a, d, z), within the time bound T in its real attack against the signer of public-key U, then with probability at least $\varepsilon = (\varepsilon' - (QR + R^2/2)/(q-1) - \frac{Q+2}{2^l-1})$ $\mathcal{F}$ outputs a valid signature (m, e, a, d, z) in the first run of $\mathcal{C}$ described in Figure 1 such that $\mathcal{F}$ made both $h(m) = e$ and $h(a) = d$ queries to the RO with the order of $h(m)$ *being posterior to* $h(a)$ or the order of $h(a)$ *being posterior to* $h(m)$. Without loss of generality, we assume it is the former case, i.e., the RO query $h(m)$ is posterior to $h(a)$ (the analysis of the case of $h(a)$ being posterior to $h(m)$ is similar). We have the following lemma, from which the theorem is then established.

Lemma 4.1 (divided forking lemma) *Suppose $\mathcal{F}$ produces, with probability ε, a valid signature (m, e, a, d, z) within the time bound T in the first run of $\mathcal{C}$ such that $\mathcal{F}$ made both $h(m) = e$ and $h(a) = d$ RO queries with the order $h(m)$* being posterior to *$h(a)$, then within time $T' \leq (2/\varepsilon + (\varepsilon/4Q - 2^{-l})^{-1}) \cdot T$ and with probability at least $\frac{1}{9}$, a replay of $\mathcal{F}$ outputs a valid signature (m, e', a, d, z') for $e' \neq e$.*

Proof (of Lemma 4.1). The proof of Lemma 4.1 is essentially identical to that of Lemma 2 in [15], which we re-produce here for completeness. We mention that, as in [15], although the divided forking lemma is presented here w.r.t. the challenge-divided Schnorr signature (based on the challenge-divided Schnorr's Σ-protocol for DLP), it can be directly extended and applied to signatures derived from other challenge-divided Σ-protocols.

Denote by ω the random tape of $\mathcal{F}$, and assume $\mathcal{F}$ makes at most Q RO queries $\mathcal{Q}_1, \cdots, \mathcal{Q}_Q$ (for presentation simplicity, we assume all RO queries are distinct), and denote by $\rho = (\rho_1, \cdots, \rho_Q)$ the Q RO answers. It is clear a random choice of the random function h (i.e., the RO) corresponds to a random choice of ρ.

Define $\mathcal{S}$ to be the set of (ω, h) such that $\mathcal{F}^h(\omega)$ outputs a valid signature (m, e, a, d, z) in the first run of $\mathcal{C}$, such that $\mathcal{F}$ made both $h(m)$ and $h(a)$ RO queries with the order of $h(m)$ being posterior to $h(a)$. That is, $\Pr[\mathcal{S}] = \varepsilon$. Define $Ind(\omega, h)$ to be the index of the RO query $h(m)$, i.e., $m = \mathcal{Q}_{Ind(\omega,h)}$. Define $\mathcal{S}_i$ to be the subset of $\mathcal{S}$ such that $Ind(\omega, h) = i$ for $1 \leq i \leq Q$. That is, the set $\{\mathcal{S}_1, \cdots, \mathcal{S}_Q\}$ is a partition of $\mathcal{S}$. Define $I = \{i | Pr[\mathcal{S}_i|\mathcal{S}] \geq 1/2Q\}$, i.e., $\Pr[\mathcal{S}_i | i \in I] \geq \varepsilon/2Q$. For each $i \in I$, define by h_i the restriction of h to queries of index strictly less than i, they by applying the Splitting Lemma (Lemma 1, page 12 in [15]), there exists a subset Ω_i (of $\mathcal{S}$) such that: (1) for any $(\omega, h) \in \Omega_i$, $\Pr_{h'}[(\omega, h') \in \mathcal{S}_i | h'_i = h_i] \geq \varepsilon/4Q$; (2) $\Pr[\Omega_i|\mathcal{S}_i] \geq \frac{1}{2}$. As all the subsets $\mathcal{S}_i$ are disjoint, it is calculated that $\Pr_{\omega,h}[\exists i (\omega, h) \in \Omega_i \cap \mathcal{S}_i | \mathcal{S}] \geq \frac{1}{4}$ (for more details, the reader is referred to [15]).

By the Lemma 3 (page 14) in [15], we get $\Pr[Ind(\omega, h) \in I | \mathcal{S}] \geq \frac{1}{2}$. Now, run $\mathcal{F}$ $2/\varepsilon$ times with random ω and h, with probability $1 - (1 - \varepsilon)^{2/\varepsilon} \geq \frac{4}{5}$ we get one successful pair $(\omega, h) \in \mathcal{S}$. Denote by β the index $Ind(\omega, h)$ corresponding to the successful pair. We know with probability at least $\frac{1}{4}$, $\beta \in I$ and $(\omega, h) \in \mathcal{S}_\beta \cap \Omega_\beta$. Consequently, with probability at least $\frac{1}{5}$, the $2/\varepsilon$ runs have provided a successful pair $(\omega, h) \in \mathcal{S}_\beta \cap \Omega_\beta$ where $\beta = Ind(\omega, h)$. As $\Pr_{h'}[(\omega, h') \in \mathcal{S}_\beta | h'_\beta = h_\beta] \geq \varepsilon/4Q$ in this case, we get $\Pr_{h'}[(\omega, h') \in \mathcal{S}_\beta \wedge \rho_\beta \neq \rho'_\beta | h'_\beta = h_\beta] \geq \varepsilon/4Q - 2^{-l}$, where $\rho_\beta = h(\mathcal{Q}_\beta)$ and $\rho'_\beta = h'(\mathcal{Q}_\beta)$. Now, we replay $\mathcal{F}$ with fixed ω but randomly chose h' such that $h'_\beta = h_\beta$, for $(\varepsilon/4Q - 2^{-l})^{-1}$ times, with probability at least $\frac{3}{5}$, we will get another success. That is, after less than $2/\varepsilon + (\varepsilon/4Q - 2^{-l})^{-1}$ repetitions of $\mathcal{F}$'s attack, with probability at lease $\frac{1}{5} \times \frac{3}{5} \geq \frac{1}{9}$, we have obtained two valid signatures (m, e, a, d, z) and (m, e', a, d, z') for $e \neq e'$. □ □

Divided forking lemma vs. standard forking lemma. The standard forking lemma proved in [15] deals with a single rewinding point (specifically, $e = h(a, m)$) in the RO model for formally proving the security of digital signatures instantiated via the standard Fiat-Shamir paradigm. This standard forking lemma does not directly apply to the above analysis of challenge-divided Schnorr signature, as we need to deal with multiple ordered rewinding points (specifically, $h(m)$ and $h(a)$) in the RO model. We suggest that this divided forking lemma can have independent interests, and can be applied to analyze other cryptographic schemes in the random oracle model, where multiple ordered rewinding points need to be dealt with in the random oracle model.

4.1 Challenge-Divided Schnorr Signature vs. DSS

We note all performance advantages of DSS are essentially preserved with the challenge-divided Schnorr scheme. We also note the techniques proposed in [14] for improving the performance of DSS in certain scenarios, e.g., signature batch verification and compression, etc, are also applicable to challenge-divided Schnorr. In addition, challenge-divided Schnorr has the following advantages over DSS:

- Provable security in the random oracle model. We have shown that, assuming both h and f are random oracles, the challenge-divided Schnorr scheme is existentially unforgeable against adaptive chosen message attacks under the DLP assumption in the random oracle model. The provable security of DSS is still unknown.

- Same or better offline space complexity than DSS. Suppose k values of a's are pre-computed, the offline space complexity of challenge-divided Schnorr with $z = er + dw$ is $3k|q|$ (which is the same as that of DSS). But, for challenge-divided Schnorr with $z = dr + ew$, the offline space complexity is only $2k|q|$.
- More efficient signature generation in total. To compute the value s in the DSS signature, the DSS signer performs 1 modular inverse (i.e., $\hat{r} = r^{-1}$) and 2 modular multiplications in total. In comparison, to compute the value z in the challenge-divided Schnorr signature, the signer only performs 2 modular multiplications in total (without performing the quite time-consuming modular inverse operation). We remark that modular inverse is a relatively expensive operation (which is typically performed by the Euclid algorithm), and is thus much preferable to dispense with (particularly for smart-card-based deployment).
- More efficient offline pre-computation. Besides the same other pre-computations, the DSS signer needs to perform 1 modular inverse r^{-1} and 2 modular multiplications for computing dwr^{-1}, but the signer of challenge-divided Schnorr needs to offline perform only 1 modular multiplication dw or dr.
- More efficient online signature verification (for the case of $z = er + dw$). For verifying a DSS signature (d, s), the verifier has to compute $\hat{s} = s^{-1}$ online (which is a relatively expensive operation as clarified), as the value s is known to the verifier *only when the signature comes to it*. In comparison, for verification of challenge-divided Schnorr with $z = er + dw$, the verifier only needs to compute the inverse $\hat{e} = e^{-1}$ where $e = h(m)$. In case the verifier learns the message to be signed prior to receiving the signature from the signer (which is quite common in certain scenarios), the values e and e^{-1} can both be offline pre-computed by the verifier of challenge-divided Schnorr. For challenge-divided Schnorr with $z = dr + ew$, signature verification is of the same computational complexity as that of DSS.

4.2 Challenge-Divided Schnorr Signature vs. Schnorr's Signature

For implementation of Schnorr' signature over (the subgroup of order q of) Z_p^*, where typically p is of 1024 bits and q 160 bits, and supposing the signer pre-computes k values of a, the space complexity (of storing k pre-computed a's) of Schnorr's signature is $(|p| + |q|)k$, which is significantly larger than that of challenge-divided Schnorr signature (that is $2|q|k$ for the case of $z = dr + ew$).

For implementations of Schnorr's signature and challenge-divided Schnorr signature on certain elliptic curves over finite field F_q, the value a is an elliptic curve point that is typically represented by a pair of coordinates $(x, y) \in F_q^2$. In this case, the space complexity (for storing k pre-computed a's) of Schnorr's signature is $3|q|k$ that is still much larger than the complexity of $2|q|k$ of challenge-divided Schnorr signature. We also note that the presentation of an elliptic curve point can be made more concise, e.g., by just using the x-coordinate [2]. But, with such a condensed presentation of elliptic curve point, the signer needs to recover the y-coordinate when generating signatures (i.e, the value $e = h(a, m)$) *in the online phase*, which can incur much additional online computational complexity and may violate the spirit of online/offline signatures. In this sense, challenge-divided Schnorr signature still enjoys better online/offline efficiency than Schnorr's signature even implemented over elliptic curves (besides significant online/offline efficiency advantage for implementations over Z_p^*).

References

[1] M. Bellare and P. Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In*ACM Conference on Computer and Communications Security*, pages 62-73, 1993.

[2] D. Boneh, B. Lynn and H. Shacham. Short Signatures from the Weil Pairing. In *AsiaCrypt 2001*, pages 514-532, LNCS 2248, Springer-Verlag, 2001.

[3] R. Cramer. Modular Design of Secure, yet Practical Cryptographic Protocols, PhD Thesis, University of Amsterdam, 1996.

[4] S. Even, O. Goldreich and S. Micali. On-line/Off-line Digital Sigantures. In Crypto'89, pages 263-277.

[5] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *A. Odlyzko (Ed.): Advances in Cryptology-Proceedings of CRYPTO'86, LNCS 263*, pages 186-194. Springer-Verlag, 1986.

[6] FIPS Pub 186-2, *Digital Signature Standard (DSS)*, Federal Information Processing Standards Publication 186-2, US Department of Commerce/National Institute of Standard and Technology, Githersburg, Maryland, USA, January 27, 2000. (Change notice is made on October 5 2001.)

[7] O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing But Their Validity or All languages in $\mathcal{NP}$ Have Zero-Knowledge Proof Systems. *Journal of the Association for Computing Machinery*, 38(1): 691-729, 1991.

[8] S. Goldwasser, S. Micali and C. Rackoff. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SIMA Journal on Computing*, 17(2): 281-308, 1988.

[9] L. Guillou and J. J. Quisquater. A Practical Zero-Knowledge Protocol Fitted to Security Microprocessor Minimizing both Transmission and Memory. In *C. G. Gnther (Ed.): Advances in Cryptology-Proceedings of EUROCRYPT 1988, LNCS 330* , pages 123-128, Springer-Verlag, 1988.

[10] H. Krawczyk. HMQV: A High-Performance Secure Diffie-Hellman Protocol. In *V. Shoup (Ed.): Advances in Cryptology-Proceedings of CRYPTO 2005, LNCS 3621*, pages 546-566. Springer-Verlag, 2005.

[11] L. Law, A. Menezes, M. Qu, J. Solinas and S. Vanstone. An Efficient Protocol for Authenticated Key Agreement. *Designs, Codes and Cryptography*, 28: 119-134, 2003.

[12] A. Menezes, M. Qu, and S. Vanstone. Some New Key Agreement Protocols Providing Mutual Implicit Authentication. Second Workshop on Selected Areas in Cryptography (SAC'95), 1995.

[13] A. Menezes and B. Ustaoglu. On the Importance of Public-Key Validation in the MQV and HMQV Key Agreement Protocols. *INDOCRYPT* 2006: 133-147.

[14] D. Naccache, D. M'Raihi, S. Vaudenay and D. Raphaeli. Can D.S.A be Improved? Complexity Trade-Offs with the Digital Signature Standard. In *Advances in Cryptology-Proceedings of EUROCRYPT 1994, LNCS 950*, pages 77-85, Springer-Verlag, 1994.

[15] D. Pointcheval and J. Stern. Security Arguments for Digital Signatures and Blind Signatures. *Journal of Cryptology*, 13: 361-396, 2000.

[16] C. Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3): 161-174, 1991.

[17] A. Shamir and Y. Tauman. Improved Online/Offline Signature Schemes. In In *Advances in Cryptology-Proceedings of CRYPTO 2001, LNCS 2139*, pages 355-367, Springer-Verlag, 1996.