[Published in Topics in Cryptology – CT-RSA 2022, Cryptographers' Track at the RSA Conference 2022 Proceedings, Steven D. Galbraith Editor]

A Pairing-Free Signature Scheme from Correlation Intractable Hash Function and Strong Diffie-Hellman Assumption

Benoît Chevallier-Mames

Zama,
41 Boulevard Malherbes,
75008 Paris, France
`benoit.chevalliermames@zama.ai`
https://zama.ai

Abstract. Goh and Jarecki (Eurocrypt 2003) showed how to get a signature scheme from the computational Diffie-Hellman assumption, and they introduced the name EDL for signatures of this type. The corresponding EDL family of signature schemes is remarkable for several reasons: elegance, simplicity and tight security. However, EDL security proofs stand in the random oracle model, and, to the best of our knowledge, extending this family without using an idealization of hash functions has never been successful.

In this paper, we propose a new signature scheme belonging to the EDL family, which is simple, natural and efficient, without using the random oracle model. Our scheme is based on the very same assumption than the Boneh-Boyen scheme, namely the strong Diffie-Hellman assumption, with the precision that our groups are not bound to being bilinear. We also make use of a correlation-intractable hash function, for a particular relation related to discrete-logarithm.

In addition to the theoretical interest of extending the EDL family without the random oracle model, our scheme is also one of the very few schemes which achieve discrete-log security properties without relying on pairings.

Keywords: Public-key cryptography · Signature schemes · Standard model · Random oracle model · Correlation intractability · Discrete logarithm problem · Diffie-Hellman problem · EDL signature scheme · Boneh-Boyen signature scheme.

1 Introduction

Signature schemes have existed since the introduction of public key cryptography [26] and constitute one of the most essential tools in the cryptographic toolbox. Among the various families of schemes that prevail, the so-called on-line/off-line signatures have the nice property that one can precompute a part of the signature beforehand, independently from the signed message, and complete the signature generation very quickly once the message is known. Most signature schemes derived from the Fiat-Shamir heuristic [31] are inherently on-line/off-line signature schemes. These schemes are of particular interest in implementations, especially in constrained environments, and have numerous applications. Fortunately, there exist generic constructions that one can use to transform any signature scheme into an on-line/off-line scheme [66] based on chameleon hashing [52].

Before the advancement of (reductionist) provable security, designing a seemingly secure signature scheme was mainly an art, but with the steady accumulation of knowledge, security definitions and constructions [41, 42] have been more and more precise, up to the point where the security of a signature scheme can be formally proven under the assumption that some algorithmic problem is hard to solve [67]. Classically, provably secure signatures divide into two categories: the ones based on integer factoring such as RSA [62, 61], and the ones based on the discrete log and Diffie-Hellman problems [30, 64]. A fundamental breakthrough in security proving has been the now widely adopted Random Oracle (RO) model [4, 5], a paradigm that models hash functions used in schemes as randomly chosen functions. The RO model made it possible to prove the security of discrete-log based signature schemes such as Schnorr under the discrete-log assumption thanks to the so-called forking lemma technique [59]. A critical feature of security proofs, however, resides in the "quality" of the reduction that goes along with the proof. A security reduction is said to be tight when its success probability is close to the adversary's success probability, otherwise it is said to be loose [54]. Carefully assessing the tightness of a reduction allows to give concrete security parameters for which the proof results in a meaningful security assurance [5, 20, 65].

The fact that RO-based security proofs do not imply that a scheme is secure in the real world is known for a long time [14, 40, 15]. Other works have shown that separations between the RO and the standard models also apply on non-pathological signature schemes such as Schnorr [58] and RSA [27, 57]. These limitations as well as the wide adoption of pairings [49, 2, 9, 10] in cryptographic design have empowered the search for new schemes that achieve provable security in the standard model. This has resulted in the appearing of new intractability assumptions such as the strong RSA assumption [36, 25] or the strong Diffie-Hellman assumption on bilinear groups [6].

Even though numerous signature schemes are known today which security is proven either in the RO model [43, 37, 60, 10, 70] or[1] in the standard model [29, 23, 11, 12] (see also [32, 53, 19, 46, 47, 45]), it is striking to observe that, excluding the use of pairings, very few discrete-log-based signature schemes were proven secure in the standard model. This includes notably the generic construction of signature schemes from one-way functions [56, 63], [22] (whose signature size grows logarithmically with the number of signatures) and the generic trans-

[1] We remark that other hash function idealizations or instantiations have also been investigated, *e.g.,* in [34, 45].

formation to turn an identity-based encryption (IBE) scheme into a signature scheme [8, 69] with pairing-free but inefficient IBE proposed in [28]. We can also list more recent schemes as [33, 1] with their proofs based on the DDH assumption in the *non-programmable* random oracle model.

Contribution. This paper introduces a new, pairing-free signature scheme, which extends the EDL scheme and its variants [16, 48, 38, 51, 18, 39] which are proven under the computational Diffie-Hellman assumption in the random oracle model. For our construction, we notably borrow ideas to Boneh-Boyen signature scheme [6, 7], whose security proof is based on the strong Diffie-Hellman (SDH) assumption in the standard model.

At a high-level view, the basic idea behind our construction is to prove that some group element h is a solution of the SDH problem, *i.e.,* that

$$h = g^{\frac{1}{x+\alpha}}$$

for known g, α and secret (but committed) x. On its own, this part is similar to the Boneh-Boyen scheme. However, while Boneh-Boyen signatures use a bilinear map to prove that h is well-formed, we rather use proofs of equality of discrete logs for the same purpose, as other schemes in the EDL family. As explained further in our security proof, we ensure equality of discrete logs, unless a very unlikely type of collision (that we link to the notion of *correlation-intractability* [14]) is found within one of the hash functions, which is typical to EDL proofs.

We mention that our scheme runs two independent instances of a simpler scheme and is therefore reminiscent to twin signatures [55] even though the final signature is obtained in a specific way. Previous schemes like [25] have also used the principle of a double key and (somehow) a signature which is like the concatenation of two independent-key related underlying signatures. Finally, our scheme has strong connections with OR-proofs [24, 33, 1], since our security proof is essentially a proof of discrete-log equality in at least one of the two independent instances.

Road-map. The rest of this paper is organized as follows: the next section provides some background on signatures and intractability assumptions. Section 3 describes the EDL and Boneh-Boyen signature schemes, as well as other relevant state of the art regarding pairing-free discrete-log signature schemes. Section 4 is the core of our paper: we describe our new signature scheme and provide a tight security proof based on the strong Diffie-Hellman assumption and the hash function intractability. We finally conclude in Section 5.

2 Definitions

This section provides some rather informal background on signature schemes and security notions attached to these. Most of these definitions are classical.

We also define the intractability assumptions on which are based the security proofs of the EDL and Boneh-Boyen signature schemes.

2.1 Signature Schemes

A signature scheme $\mathsf{Sig} = (\mathsf{SetUp}, \mathsf{GenKey}, \mathsf{Sign}, \mathsf{Verify})$ is defined by the four following algorithms:

- The *set-up algorithm* SetUp. On input 1^k, algorithm SetUp produces some common parameters.
- The *key generation algorithm* GenKey. On input 1^k, algorithm GenKey produces a pair $(\mathsf{pk}, \mathsf{sk})$ of matching public (verification) and private (signing) keys.
- The *signing algorithm* Sign. Given a message m in a set of messages $\mathcal{M}$ and a pair of matching public and private keys $(\mathsf{pk}, \mathsf{sk})$, Sign produces a signature σ. Often, the signing algorithm is probabilistic.
- The *verification algorithm* Verify. Given a signature σ, a message $m \in \mathcal{M}$ and a public key pk, Verify tests whether σ is a valid signature of m with respect to pk.

Several security notions have been defined about signature schemes, mainly based on the seminal work of Goldwasser, Micali and Rivest [41, 42]. It is now customary to ask for the impossibility of existential forgeries, even against adaptive chosen-message adversaries:

- An *existential forgery* is a signature, valid and generated by the adversary, corresponding to a message which was never submitted to the signature oracle. The corresponding security notion is called *existential unforgeability* (EUF).
- The verification key is public, including to the adversary. But more information may also be available. The strongest kind of information is definitely formalized by the *adaptive chosen-message attacks* (CMA), where the attacker can ask the signer to sign any message of its choice, in an adaptive way.

As a consequence, we say that a signature scheme is *secure* if it prevents existential forgeries, even under adaptive chosen-message attacks ($\mathsf{EUF\text{-}CMA}$). This is measured by the following success probability, which should be negligibly small, for any adversary $\mathcal{A}$ which outputs a valid signature σ on a message m that was never submitted to the signature oracle, within a “reasonable” bounded running-time and with at most q_s signature queries to the signature oracle:

$$\mathsf{Succ}_{\mathsf{Sig}}^{\mathsf{EUF-CMA}}(\mathcal{A}, q_s) = \Pr\left[\begin{array}{c}(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{GenKey}(1^k), (m, \sigma) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk};\cdot)}(\mathsf{pk}) : \\ \mathsf{Verify}(\mathsf{pk}; m, \sigma) = \textsc{True}\end{array}\right] .$$

When the signature generation is not deterministic, several signatures may correspond to the same message. In this case, we do not consider the attacker successful when it outputs a second signature on a message already submitted to the signature oracle. Being given a message-signature pair (m, σ), providing a second signature σ' on the same message m is captured by the adversarial goal of *malleability* [68].

2.2 Intractability Assumptions

Let us consider a cyclic (multiplicatively written) group $\mathbb{G}$ of prime order q generated by g, *i.e.*, $\mathbb{G} = \{g^i, i \in \mathbb{Z}_q\}$.

DL. Solving the discrete log problem in $\mathbb{G}$ consists, given g and $y = g^x$ for some unknown random integer $x \in \mathbb{Z}_q$, in finding the value of $\mathsf{DL}_g(y) = x$.

CDH. Solving the (computational) Diffie-Hellman (CDH) problem is as follows:[2] given (g, g^a, g^x) for unknown random integers $a, x \in \mathbb{Z}_q$, one must compute the group element g^{ax}.

q-SDH (bilinear setting) [6, 7]. Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be three multiplicative groups of prime order q and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ an admissible bilinear map [10, 3, 44, 35]. The q-strong Diffie-Hellman problem (q-SDH) consists in computing a pair

$$(\alpha, {g_2}^{\frac{1}{x+\alpha}})$$

given a $(q+2)$-tuple $(g_1, {g_1}^x, ..., {g_1}^{x^i}, ..., {g_1}^{x^q}, g_2, {g_2}^x)$, for a random integer $x \in \mathbb{Z}_q$, generators g_1 and g_2 of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively, and a security parameter q. The q-SDH problem on bilinear groups was further studied in [17, 7]. Note that $g_2, {g_2}^x$ may be omitted in the input tuple when $\mathbb{G}_1 = \mathbb{G}_2$.

In this work, we make use of the q-SDH in a general, non-bilinear context.

q-SDH (general setting) [6, 7]. Solving q-SDH in the general setting consists in computing a pair

$$(\alpha, g^{\frac{1}{x+\alpha}})$$

given a q-tuple $(g, g^x, ..., g^{x^i}, ..., g^{x^q})$, for a random integer $x \in \mathbb{Z}_q$ and a security parameter q. Note that deciding whether a solution for the bilinear q-SDH problem is valid is easy since the pairing provides a straightforward way to verify it. This is however not the case in the general setting, meaning that q-SDH admits a non-trivial decisional variant.

In particular, q-SDH in the general setting is *non-falsifiable.* However, as shown in Section 4.4, with auxiliary information (the (s_i, c_i)'s of the scheme described in Section 4.2), one can efficiently decide if a pair is a valid q-SDH pair.

[2] Whether DL and CDH are equivalent is actually an open question.

3 Prior Art

We now review signatures schemes that enjoy a tight EUF-CMA security under the discrete-log-related complexity assumptions discussed above. In this section, ℓ_p, ℓ_q, and ℓ_r denote security parameters. The schemes make use of cyclic groups $\mathbb{G}$ (resp. $\mathbb{G}_1$ and $\mathbb{G}_2$) of prime order q generated by g (resp. g_1 and g_2) where q is a ℓ_q-bit prime. We assume that elements of $\mathbb{G}$ (resp. $\mathbb{G}_1$ and $\mathbb{G}_2$) can be represented as binary strings in $\{0,1\}^{\ell_p}$. The set of messages to be signed is denoted $\mathcal{M}$.

3.1 The EDL Family of Signatures

The EDL[3] signature scheme, independently proposed in [16, 48], is defined as follows.

Set-up: Let two hash functions, $\mathcal{H} : \mathcal{M} \times \{0,1\}^{\ell_r} \to \mathbb{G}$ and $\mathcal{G} : \mathbb{G}^6 \to \mathbb{Z}_q$.

Key generation: The private key is a random number $x \in \mathbb{Z}_q$. The corresponding public key is $y = g^x$.

Signature: To sign a message $m \in \mathcal{M}$, one first randomly chooses $r \in \{0,1\}^{\ell_r}$, and computes $h = \mathcal{H}(m, r)$ and $z = h^x$. Follows a proof of logarithm equality that $\mathsf{DL}_h(z) = \mathsf{DL}_g(y)$: for a random number $k \in \mathbb{Z}_q$, one computes $u = g^k$, $v = h^k$, $c = \mathcal{G}(g, h, y, z, u, v)$ and $s = k + cx \bmod q$. The signature on m is $\sigma = (z, r, s, c)$.

Verification: To verify a signature $\sigma = (z, r, s, c) \in \mathbb{G} \times \{0,1\}^{\ell_r} \times \mathbb{Z}_q^2$ on a message $m \in \mathcal{M}$, one computes $h = \mathcal{H}(m, r)$, $u = g^s\, y^{-c}$ and $v = h^s\, z^{-c}$. The signature σ is accepted iff $c = \mathcal{G}(g, h, y, z, u, v)$.

In the random oracle model, the chosen-message security of EDL reduces to the security of the computational Diffie-Hellman problem [38], by showing that the EDL scheme is a proof that $\mathsf{DL}_h(z) = \mathsf{DL}_g(y) = x$. The scheme yields signatures of $(\ell_p + 2\ell_q + \ell_r)$ bits (for typical setting, $\ell_r = 111$). In its classical use, the scheme cannot be used with precomputations (*a.k.a.* coupons) before knowing the message, but, as noted by Goh and Jarecki, one can use the technique of [66] based on chameleon hash functions [52] to transform this signature into a signature with coupons, at the price of larger signatures.

The Katz-Wang Variants. In [51][4], Katz and Wang proposed two variants of EDL, one based on DDH assumption, and the other which yields shorter signatures still with tight relation to the CDH problem.

[3] The name EDL was proposed in [38], based upon the fact that the scheme is a proof of equality of discrete-logarithms.

[4] A remarkable unification of [38, 51] papers appeared in [39].

Being relatively generic to signature schemes with randomness,[5] the idea of Katz and Wang is to remove the *randomness* of r, and to replace it by *unpredictability*. Namely, r is replaced by a bit b that can only be computed by the signer (*e.g.,* b is the result of a pseudo-random function, under a secret key included in the signing key):[6] the signatures are then (z, b, s, c), and so are shorter than EDL signatures by 110 bits. This modification gives a signature scheme with a signature length of $(\ell_p + 2\ell_q + 1)$ bits. In this scheme, as in EDL, only u can be computed off-line, and so the on-line part of the signature is two modular exponentiations in $\mathbb{G}$.

The Chevallier-Mames Variant. In [18], another EDL variant was proposed. The main modification is how the value h is computed. Instead of being $h = \mathcal{H}(m, r)$ as in EDL or $h = \mathcal{H}(m, b)$ as in Katz-Wang scheme, one sets $h = \mathcal{H}(u)$.

Set-up: Let two hash functions, $\mathcal{H} : \mathbb{G} \to \mathbb{G}$ and $\mathcal{G} : \mathcal{M} \times \mathbb{G}^6 \to \mathbb{Z}_q$.

Key generation: The private key is a random number $x \in \mathbb{Z}_q$, while the corresponding public key is $y = g^x$.

Signature: To sign a message $m \in \mathcal{M}$, one first randomly chooses $k \in \mathbb{Z}_q$, and computes $u = g^k$, $h = \mathcal{H}(u)$, $z = h^x$ and $v = h^k$. Next, one computes $c = \mathcal{G}(m, g, h, y, z, u, v)$ and $s = k + cx \bmod q$. The signature on m is $\sigma = (z, s, c)$.

Verification: To verify a signature $\sigma = (z, s, c) \in \mathbb{G} \times \mathbb{Z}_q^2$ on a message $m \in \mathcal{M}$, one computes $u = g^s\, y^{-c}$, $h = \mathcal{H}(u)$, and $v = h^s\, z^{-c}$. The signature σ is accepted iff $c = \mathcal{G}(m, g, h, y, z, u, v)$.

The signatures are a little bit smaller than the EDL's ones: they are only $(\ell_p + 2\ell_q)$-bit long. Interestingly, this scheme natively allows on-line/off-line signature scheme, *i.e.,* without affecting the efficiency of the signature or of the verification nor the signature size. The scheme remains tightly related to the computational Diffie-Hellman problem, still in the random oracle model.

Proving equality of discrete logs. A common part of the respective proofs of the previous schemes (see [39, 18] for more details) consists in showing that it is very unlikely that the forger can provide a valid signature with $u = g^k$, $v = h^{k'}$ and $z = h^{x'}$ with $k \neq k'$ or $x \neq x'$. More precisely, the forger would need to find (m, k, k', x') such that[7]

$$c = \mathcal{G}(m, g, h, y, z, u, v) = \mathcal{G}(m, g, h, y, h^{x'}, g^k, h^{k'}) = \frac{k - k'}{x' - x} \bmod q$$

[5] Notably, the very same idea can be applied on Probabilistic RSA-FDH to get a tight signature scheme with a single extra bit.

[6] In other words, in EDL, signing few times the same message would result in different random numbers r, while doing the same with Katz-Wang scheme would give always the same bit b.

[7] Strictly, the equality is for Chevallier-Mames variant; for EDL or Katz-Wang, m is not an input of $\mathcal{G}$, without changing anything to the analysis.

which is doable only with a probability $\frac{q_{\mathcal{G}}}{q}$ if $\mathcal{G}$ is assumed to be a random oracle and $q_{\mathcal{G}}$ is the number of requests to $\mathcal{G}$ oracle.

Remark that previous equation can be rewritten as follows, if one defines $\beta = \mathsf{DL}_g(h)$:

$$c = \mathcal{G}(m, g, g^{\beta}, g^{x}, g^{\beta x'}, g^{k}, g^{\beta k'}) = \frac{k - k'}{x' - x} \bmod q$$

which can also be seen as

$$\mathcal{G}(m, g, g^{a_1}, g^{a_2}, g^{a_3}, g^{a_4}, g^{a_5}) = \frac{a_4 - a_5/a_1}{a_3/a_1 - a_2} = \frac{a_1 a_4 - a_5}{a_3 - a_1 a_2} \bmod q$$

In our scheme, we formalize the notion of *discrete-log collision resistant* hash function (in Section 4.3) and use it to prove the security of our scheme (in Section 4.4).

3.2 Boneh-Boyen Signatures

In [6, 7], a completely different way to prove that a tuple is a Diffie-Hellman tuple was used: the pairing.

Set-up: Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be three groups whose prime order is q, for which it exists an efficient pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$.

Key generation: Let g_1 and g_2 be two random generators of $\mathbb{G}_1$ and $\mathbb{G}_2$. Let x, y be two random integers of $\mathbb{Z}_q$. Let $u = {g_2}^x$, $v = {g_2}^y$, $z = e(g_1, g_2)$. The private key is (x, y, g_1) and the corresponding public key is (g_2, u, v, z).

Signature: To sign a message $m \in \mathbb{Z}_q$, one first randomly chooses $r \in \mathbb{Z}_q$ ($r \neq -\frac{x+m}{y}$), and computes $s = {g_1}^{\frac{1}{x+m+y\cdot r}}$. The signature on m is $\sigma = (s, r)$.

Verification: To verify a signature $\sigma = (s, r) \in \mathbb{G}_1 \times \mathbb{Z}_q$ on a message $m \in \mathcal{M}$, one checks that $e(s, u \cdot {g_2}^m \cdot v^r) = z$. If true, the signature σ is accepted.

Boneh-Boyen signature scheme has the following notable differences with EDL variants: its security is based on q_s-SDH problem (where q_s is the number of signature queries), and does not require the random oracle model. A security proof in the standard model rather than the RO model is arguably an important improvement over pre-existing schemes. However, we stress that the result was mainly achieved thanks to the use of pairings. Evaluating a pairing at verification time may result in a slower and more complicated implementation as opposed to when only common group operations are performed.

3.3 Existing Pairing-Free Discrete-log Signature Schemes in the Standard Model

In the set of signature schemes provably secure under a discrete-log-kind assumption, it is remarkable that most schemes rely on pairing. We succinctly describe here two schemes which are pairing-free but significantly less efficient than the scheme we propose in Section 4.2.

First, it is a classical result that signature schemes can be generically built from one-way functions [56, 63]. This type of construction is however particularly inefficient in term of signature size.

Cramer-Damgård scheme In [22], Cramer and Damgård have shown a generic construction, which can be instantiated for the discrete-log case as described in their Section 6. The principle relies on the following zero-knowledge protocol.

Key generation: Let d be a security parameter. Let x_i be random elements of $\mathbb{Z}_q$, and $y_i = g^{x_i}$, for $i \in \{0, ..., d-1\}$. The private key is $\{x_i\}$ and the corresponding public key is $\{y_i\}$.

Commit: The prover generates a random $k \in \mathbb{Z}_q$ and sends $u = g^k$ to the verifier.

Challenge: The verifier generates random $c_i \in \mathbb{Z}_q$ for $i \in \{0, ..., d-1\}$, and sends them to the prover.

Answer: The prover computes $s = k + \sum_{i=0}^{d-1} c_i \cdot x_i \bmod q$ and sends s to the verifier.

Verification: Finally, the verifier checks whether $g^s = u \cdot \prod_{i=0}^{d-1} {y_i}^{c_i}$.

With the use of an authentication tree, Cramer and Damgård get a signature scheme whose signature size is $\mathcal{O}(\ell_p \cdot \mathsf{log}(q_s))$. The number of exponentiations to compute or verify the signature also depends on the depth of the tree.

Generic transformation from an identity-based encryption scheme into a signature scheme Another possibility to have a scheme based on a discrete-log assumption without relying on pairings is to use the generic transformation from an identity-based encryption scheme into a signature scheme, that was proposed by Boneh and Franklin in seminal [8] (see [69] as well). Regarding the purpose of this paper, this generic transformation can be combined with pairing-free scheme proposed in [28] to get a signature scheme. However, as remarked by Döttling and Garg, the scheme is particularly inefficient.

3.4 OR-based Signature Schemes

OR-proofs [24] are one of the fascinating techniques used to achieve security proofs, which was notably recently used to achieve signatures schemes as in [33, 1]. In this section, we remind the reader of the first construction, which can notably be instantiated to get a scheme based on the DDH assumption in the non-programmable random oracle, in a scheme comparable to ours.

Fischlin, Harasser and Janson scheme The principle is to have two Sides 0 and 1, with their respective DDH tuple $(y_i, h_i, z_i) = (g^{x_i}, g^{a_i}, g^{a_i\,x_i})$. One Side $b \in \{0, 1\}$ is the preferred side. Roughly, the non-preferred side will be completely simulated (and so x_{1-b} is not needed in the signature processing), while the Side b will go in a process very close to Katz-Wang DDH scheme [51]. However, for the security of the construction, the role of the two sides remains completely symmetric, such that they are non distinguishable.

Set-up: Let a hash function $\mathcal{H} : \{0,1\} \times \mathbb{G}^7 \times \mathbb{G}^2 \times \mathcal{M} \to \mathbb{Z}_q$.

Key generation: Let b be a random bit and $\tilde{b} = 1 - b$. Let g and h_b be two generators of $\mathbb{G}$ and x_b be a scalar of $\mathbb{Z}_q$. Let $(y_b, z_b) = (g^{x_b}, {h_b}^{x_b})$. Finally, let $(g, y_{\tilde{b}}, h_{\tilde{b}}, z_{\tilde{b}})$ be another random DDH tuple. The private key is (b, x_b) while the public key is $\mathrm{pk} = (g, y_0, y_1, h_0, h_1, z_0, z_1)$.

Signature: To sign a message $m \in \mathcal{M}$, one first pick a random $k_b \in \mathbb{Z}_q$, then compute $u_b = {g_b}^{k_b}$ and $v_b = {h_b}^{k_b}$, which is the commitment of the Side b. Then, the Side $\tilde{b}$ is completely simulated by picking a random $s_{\tilde{b}} \in \mathbb{Z}_q$, and computing $c_{\tilde{b}} = \mathcal{H}(b, \mathrm{pk}, u_b, v_b, m)$, $u_{\tilde{b}} = {g_{\tilde{b}}}^{s_{\tilde{b}}}\, {y_{\tilde{b}}}^{-c_{\tilde{b}}}$ and $v_{\tilde{b}} = {h_{\tilde{b}}}^{s_{\tilde{b}}}\, {z_{\tilde{b}}}^{-c_{\tilde{b}}}$. Finally, the Side b is completed, by computing $c_b = \mathcal{H}(\tilde{b}, \mathrm{pk}, u_{\tilde{b}}, v_{\tilde{b}}, m)$ and $s_b = k_b + c_b x_b \bmod q$. The signature is $\sigma = (s_0, s_1, c_0, c_1)$.

Verification: To verify a signature $\sigma = (s_0, s_1, c_0, c_1) \in \mathbb{Z}_q^4$ on a message $m \in \mathcal{M}$, one computes $u_0 = {g_0}^{s_0}\, {y_0}^{-c_0}$, $u_1 = {g_1}^{s_1}\, {y_1}^{-c_1}$, $v_0 = {h_0}^{s_0}\, {z_0}^{-c_0}$ and $v_1 = {h_1}^{s_1}\, {z_1}^{-c_1}$. The signature is accepted iff $c_0 = \mathcal{H}(1, \mathrm{pk}, u_1, v_1, m)$ and $c_1 = \mathcal{H}(0, \mathrm{pk}, u_0, v_0, m)$.

The fundamental principle in this construction is that the commitment (u, v) used in the computation of c is the commitment from the other side, which allows the author of [33] to prove that their scheme has a tight security on the DDH in the non-programmable random oracle model.

Regarding efficiency, we can note that the signature size is $4\ell_q$ and that some parts of the computations (more precisely u_b and v_b) can be done before the message is known. Let us remark however that it is also possible for the signer to know the discrete logarithm $x_{\tilde{b}}$ as well, not to have to simulate the Side $\tilde{b}$, in order to be able to precompute $u_{\tilde{b}}$ and $v_{\tilde{b}}$ as well.

4 Our Signature Scheme

4.1 Intuition of the Design

In this section, we explain how we finally came to our design, to explain its difference with previous state of the art, and why we had to do these changes.

From the beginning, we wanted to extend the EDL family described in Section 3.1 to the standard model. However, this will is blocked in its early steps by the fact that, in these schemes, h is the output of the hash function $\mathcal{H}$ — although the exact form of h is different from one scheme to the other — and that there is some h^x to compute: without the random oracle model, it seems impossible to compute such quantities during the signature queries, unless by having the secret key x. In other words, it seems at first glance that $\mathcal{H}$ *has* to be a random oracle to allow the security reduction to compute pairs of the form (h, h^x) without key x.

Inversing the problem. To be able to achieve security without the random oracle model (and notably, the programmability property), we had to fundamentally change the design, and notably, we somehow *inversed the problem*, by — let's say — "making h hard to compute" and "z simple to deduce". For this, we borrow ideas from [6] and notably make use of a q_s-SDH instance to essentially replace the random oracle.

To this end, we now pose $h = g^{\frac{1}{x+\alpha}}$ for some α (which requires the secret key x to be computed) and see that $z = h^x$ is easy to compute since

$$z = g^{\frac{x}{x+\alpha}} = g \cdot h^{-\alpha}.$$

Therefore, being given h, if one can prove that z defined by $z = g \cdot h^{-\alpha}$ is such that $z = h^x$ then the well-formedness of h — *i.e.*, , the fact that $h = g^{\frac{1}{x+\alpha}}$ — is automatically proven. As should be already clear to the reader, we are using technique a-la EDL to prove that $z = h^x$.

Checking equality of logarithms. However, our goal is still not achieved: the second complicated part is to be able to prove that the two logarithms $\mathsf{DL}_g(y)$ and $\mathsf{DL}_h(z)$ are equal (to answer signature queries), but at the same time, from the signature forge, to learn something (to answer the q_s-SDH challenge).

In the random-oracle proofs, achieving these two things at the same time is easy, since one can simulate thanks to the random oracle model, and simultaneously, know a new $z = h^x$ from the forge, to solve the CDH problem. Very informally,[8] during signature queries, the random oracle allows to take random s, c and deduce u, v, h, z from these quantities, and still "close the loop", *i.e.*, make that $c = \mathcal{G}(m, g, h, u, v, y, z)$. In the standard model, on the contrary, $\mathcal{G}$ function cannot be bent to our will.

Consequently, a second major change of design in our scheme as compared to the rest of the EDL family is to have a double-key mechanism. We have two sides (see the full description in our next section), which are linked together by a new degree of liberty e. In the security proof, the reduction knows one (and only one) of the two keys, and can simulate the unknown side a bit like in the random oracle model, and complete the signature with the known key. For this aspect of the design and of the proof, we are very close to the OR-based schemes described in Section 3.4. As we will show in the security proof, the known side is indistinguishable for the adversary, and even with this new degree of liberty, the security reduction can turn the forge into an answer to the q_s-SDH challenge.

[8] The reader is referred to original papers [38, 51, 18, 39] for more formal proofs.

4.2 Description

Our scheme is made of two instances of the same flow, which we will call Side 0 and Side 1. Note that this almost entirely symmetric design presents some similarities with the approach undertaken with twin signatures [55] or other schemes as [25, 33, 1].

The signature scheme is depicted as follows.

Set-up: Let ℓ_p and ℓ_q denote security parameters. Select a cyclic group $\mathbb{G}$ of prime order q. Let $\mathbb{G}^\diamond$ be equal to $\mathbb{G}\backslash\{1\}$. Finally, select two hash functions, $\mathcal{H} : \mathbb{G}^2 \to \mathbb{Z}_q$ and $\mathcal{G} : \mathcal{M} \times \mathbb{G}^{11} \to \mathbb{Z}_q$.

Key generation: Let g, g_0 and g_1 be three random generators of $\mathbb{G}$. The private key is made of two random numbers $(x_0, x_1) \in \mathbb{Z}_q^2$. The corresponding public key is (g, g_0, g_1, y_0, y_1) with $y_0 = g^{x_0}$ and $y_1 = g^{x_1}$.

Signature: To sign a message $m \in \mathcal{M}$, randomly select $(k_0, k_1, e) \in \mathbb{Z}_q^3$, and then proceed as follows

$$\begin{array}{lcl}
\underline{\text{Side 0}} & & \underline{\text{Side 1}} \\
u_0 = g^{k_0} & & u_1 = g^{k_1} \\
 & \alpha = \mathcal{H}(u_0, u_1) & \\
h_0 = g_0{}^{\frac{1}{x_0+\alpha}} & & h_1 = g_1{}^{\frac{1}{x_1+\alpha}} \\
v_0 = h_0{}^{k_0} & & v_1 = h_1{}^{k_1} \\
z_0 = h_0{}^{x_0} & & z_1 = h_1{}^{x_1} \\
 & d = \mathcal{G}(m, g, h_0, y_0, z_0, u_0, v_0 & \\
 & \qquad h_1, y_1, z_1, u_1, v_1) & \\
c_0 = d + e \bmod q & & c_1 = -e \bmod q \\
s_0 = k_0 + c_0 \cdot x_0 \bmod q & & s_1 = k_1 + c_1 \cdot x_1 \bmod q
\end{array}$$

If $\alpha + x_0 = 0 \bmod q$ or $\alpha + x_1 = 0 \bmod q$, other randoms are picked. The signature on m is $\sigma = (h_0, s_0, c_0, h_1, s_1, c_1)$.

Verification: To verify a signature $\sigma = (h_0, s_0, c_0, h_1, s_1, c_1) \in \mathbb{G}^\diamond \times \mathbb{Z}_q^2 \times \mathbb{G}^\diamond \times \mathbb{Z}_q^2$ on a message $m \in \mathcal{M}$, one computes

$$\begin{array}{lcl}
u_0 = g^{s_0}\, y_0{}^{-c_0} & & u_1 = g^{s_1}\, y_1{}^{-c_1} \\
 & \alpha = \mathcal{H}(u_0, u_1) & \\
z_0 = g_0 \cdot h_0{}^{-\alpha} & & z_1 = g_1 \cdot h_1{}^{-\alpha} \\
v_0 = h_0{}^{s_0}\, z_0{}^{-c_0} & & v_1 = h_1{}^{s_1}\, z_1{}^{-c_1} \\
 & d = \mathcal{G}(m, g, h_0, y_0, z_0, u_0, v_0, & \\
 & \qquad h_1, y_1, z_1, u_1, v_1) &
\end{array}$$

The signature is accepted iff $c_0 + c_1 = d \pmod q$.

As shown later, signatures as per our scheme provide a proof that either $\mathsf{DL}_{h_0}(z_0) = \mathsf{DL}_g(y_0)$ or $\mathsf{DL}_{h_1}(z_1) = \mathsf{DL}_g(y_1)$ or both. One can note that this "one of the DL-equalities holds" is exactly what one needs to create a ring signature, and notably, we can see our construction as a kind of ring signature with only two virtual signers, one per side.

As explained later, we base our security proof on the property that one of the two sides can be perfectly simulated but not both at the same time.

Correctness. The scheme is consistent since, if the signature is well formed, for $\delta \in \{0, 1\}$,

$$h_\delta = {g_\delta}^{\frac{1}{x_\delta + \alpha}}, \text{ for } \alpha = \mathcal{H}(u_0, u_1).$$

It follows that $z_\delta = {h_\delta}^{x_\delta} = {g_\delta}^{\frac{x_\delta}{x_\delta+\alpha}} = {g_\delta}^{\frac{x_\delta+\alpha-\alpha}{x_\delta+\alpha}}$, and consequently $z_\delta = g_\delta \cdot {h_\delta}^{-\alpha}$.

Discussion. The main features of our scheme is that it does not rely on pairings and, at the same time, achieves chosen-message security (without the random-oracle model) with a tight reduction. Our signatures are $2\ell_p + 4\ell_q$ bits. Our construction also inherently supports on-line/off-line precomputations, *i.e.,* one can perform most of the computations *before* knowing the message m: only remains the computation of (c_0, c_1, s_0, s_1) once the message is finally known.

A comparison with schemes of Section 3 could be the following:

– as opposed to the EDL family of schemes of Section 3.1, our scheme does not need the random oracle model
– as opposed to Boneh-Boyen scheme of Section 3.2, our scheme does not use pairings
– our scheme is more efficient than generic constructions of Section 3.3.

A comparison with OR-based schemes of Section 3.4 shows that results are pretty close. [33] scheme is based on the DDH assumption, which is more classical than the SDH assumption we use in our scheme (which, in a pairing-free group, is non-falsifiable). However, SDH is a computational problem while DDH is a decisional problem. Also, even if SDH is non-falsifiable, we argue in Section 4.4 that, with the auxiliary (s_i, c_i)'s information of our scheme, one can check that whether a pair is a valid SDH pair or not. Regarding the security model, the non-programmable random-oracle model used in [33] is slightly stronger than simply assuming correlation intractability of the hash function as we do. Finally, [33] signature size is smaller than our signature size.

4.3 Introducing Discrete-Log Collisions

Let us start with a definition.

Definition 1 (Discrete-log collisions). *Let $\mathcal{G}$ be a hash function mapping tuples of $\mathcal{M} \times \mathbb{G}^{n+1}$ to $\mathbb{Z}_q$. Let $\mathcal{F}$ be a function mapping vectors of $\mathbb{Z}_q^n$ to $\mathbb{Z}_q$.*

An algorithm is said to $(\varepsilon_{\mathsf{dl}_{\mathcal{G},\mathcal{F}}}, \tau)$-break the intractability of finding discrete-log collisions of $\mathcal{G}$ with respect to $\mathcal{F}$ if, being given a fresh random generator $g \in \mathbb{G}$,[9] it can find, with a probability $\varepsilon_{\mathsf{dl}_{\mathcal{G},\mathcal{F}}}$ and within a time τ, a tuple $(m, g, g^{a_1}, g^{a_2}, ..., g^{a_n})$ such that

$$\mathcal{G}(m, g, g^{a_1}, g^{a_2}, ..., g^{a_n}) = \mathcal{F}(a_1, a_2, ..., a_n)$$

Such a tuple is called a discrete-log collision of $\mathcal{G}$ with respect to $\mathcal{F}$ and g.

Correlation-intractability This security notion for hash function is actually an instantiation of so-called *correlation-intractability.* As introduced in [14], a hash function $\mathcal{G}$ is said correlation-intractable with respect to a relation $\mathcal{R}$ if it is computationally infeasible to exhibit u such that $(u, \mathcal{G}(u)) \in \mathcal{R}$.

In the case of Definition 1, the relation $\mathcal{R}_{\mathcal{F}}$ can be defined as follows. Since g is a generator, the function

$$\begin{aligned} \Lambda\colon \mathcal{M} \times \mathbb{G}^{n+1} &\to \mathcal{M} \times \mathbb{G} \times \mathbb{Z}_q^n \\ (m, g, g^{a_1}, g^{a_2}, ..., g^{a_n}) &\mapsto (m, g, a_1, a_2, ..., a_n) \end{aligned}$$

is bijective. So, to any $u = (m, g, g^{a_1}, g^{a_2}, ..., g^{a_n})$ corresponds a unique $v = \Lambda(u) = (m, g, a_1, a_2, ..., a_n)$, and so a unique $w = (a_1, a_2, ..., a_n)$ which we note as $w = \Lambda_a(u)$. We then say that $(u, v) \in \mathcal{R}_{\mathcal{F}}$ iff $v = \mathcal{F}(\Lambda_a(u))$. Remark that saying that $(u, \mathcal{G}(u)) \in \mathcal{R}_{\mathcal{F}}$ is the same as saying that u is a discrete-log collision of $\mathcal{G}$ with respect to $\mathcal{F}$.

Building a hash function whose correlation-intractability is formally proved is out of scope of this paper.[10] Instead, in our scheme, we use a hash function $\mathcal{G}$, and show that a potential forge can be turned into showing that $\mathcal{G}$ is *not* correlation-intractable (or some other hard problem is solved, see Section 4.4).

Checking discrete-log collisions One could note that verifying that a given tuple is a valid discrete-log collision may be achieved in two ways: one way is to provide the tuple $(m, a_1, a_2, ..., a_n)$, *i.e.,* to disclose all the discrete logarithms $a_1, a_2, ..., a_n$. However, if the analytic form of $\mathcal{F}$ is simple enough (*e.g.,* our function $\mathcal{F}_0$ below), a verification that the collision is valid can be performed "in the exponents", *i.e.,* by providing proofs of equations followed by the inputs (see Section 4.4, Case 2.1).

For any hash function, having resistance against discrete-logarithm collisions is actually a desirable property, even if not a classical one. As Λ is bijective, when computing $\mathcal{G}(m, g, i_1, i_2, ..., i_n)$ for any $(m, g, i_1, i_2, ..., i_n) \in \mathcal{M} \times \mathbb{G}^{n+1}$, there is only one target value $v = \mathcal{F}(\Lambda_a(m, g, i_1, i_2, ..., i_n))$ which can lead to a discrete-log collision. Furthermore, g is picked randomly after $\mathcal{F}$ and $\mathcal{G}$ are

[9] Notably, $\mathcal{F}$ and $\mathcal{G}$ definitions cannot suppose the generator g to be already defined.
[10] One may read [50, 13, 21] for state of the art on this area of research.

defined.[11] Thus, for cryptographic hash functions $\mathcal{G}$ — notably but not only *non-programmable* random oracle hash functions [34] —, finding discrete-log collision should be hard.

Setting a particular $\mathcal{F}$ for our scheme. In this paper, we are only interested in $n = 10$ and function $\mathcal{F} = \mathcal{F}_0$ defined as follows, which is a purely modular rational function

$$\mathcal{F}_0(a_1, a_2, ..., a_{10}) = \frac{a_1a_4 - a_5}{a_3 - a_1a_2} + \frac{a_6a_9 - a_{10}}{a_8 - a_6a_7} \bmod q,$$

resulting in that a discrete-log collision can be verified by providing equations satisfied by $g^{a_1}, g^{a_2}, ..., g^{a_{10}}$ as shown in Case 2.1 of Section 4.4.

Relation to security proofs of EDL variants. Remarkably, the discrete-logarithm collision notion was already present in EDL security proofs [38, 51, 18], even if not explicitly defined (see as well the end of our Section 3.1). Notably, the authors of these papers were using $n = 5$ and

$$\mathcal{F}_1(a_1, a_2, ..., a_5) = \frac{a_1a_4 - a_5}{a_3 - a_1a_2} \bmod q.$$

4.4 Security Proof

In this section, we prove the security of our scheme under the q-SDH assumption over the group $\mathbb{G}$. Recall that $\mathbb{G}$ is an arbitrary prime-order group here, and is not required to admit a bilinear map.

Before proving the security theorem, we refer to a useful lemma whose proof can be found in [6, 7].

Lemma 1 (Proof of Lemma 9, [7]). *Let f be the polynomial*

$$f(X) = \prod_{j=1}^{j=q_s} (X + \kappa_j)$$

for some κ_j in $\mathbb{Z}_q$, and let θ be a random integer in $\mathbb{Z}_q$. Given $\{g^{x^i}\}_{i=0,\dots,q_s}$, let us define g_0 as $g_0 = g^{\theta f(x)}$. It is easy to compute g_0 and $g_0^{\frac{1}{x+\kappa_i}}$ for any $i \in [1, q_s]$. Furthermore, if one is given $h = g_0^{\frac{1}{x+\alpha}}$ with $\alpha \neq \kappa_j$, then one can easily compute $g^{\frac{1}{x+\alpha}}$.

We now state:

[11] If $\mathcal{F}$ and g were chosen by the solver, this latter could trivially pick any $(m, g, g^{a_1}, g^{a_2}, ..., g^{a_n})$, precompute $f = \mathcal{G}(m, g, g^{a_1}, g^{a_2}, ..., g^{a_n})$, and choose $\mathcal{F}(a_1, a_2, ..., a_n)$ as the constant function f.

Theorem 1. *Let $\mathcal{A}$ be an adversary against our scheme that returns an existential forgery under a chosen-message attack with success probability ε within time τ, after q_s queries to the signing oracle. Further assume that $\mathcal{H}$ and $\mathcal{G}$ are respectively $(\varepsilon_{\mathcal{H}}, \tau)$ and $(\varepsilon_{\mathcal{G}}, \tau)$-collision secure and that finding discrete-log collisions of $\mathcal{G}$ with respect to $\mathcal{F}_0$ is $(\varepsilon_{\mathsf{dl}_{\mathcal{G},\mathcal{F}_0}}, \tau)$-intractable, then the q_s-SDH problem over $\mathbb{G}$ can be solved with success probability ε' such that*

$$3\varepsilon' + 3\varepsilon_{\mathsf{dl}_{\mathcal{G},\mathcal{F}_0}} + \varepsilon_{\mathcal{G}} + \varepsilon_{\mathcal{H}} \geq \varepsilon$$

in time τ' with

$$\tau' \lesssim \tau + \mathcal{O}(q_s) \cdot \tau_0$$

where τ_0 is the time required to perform a group exponentiation in $\mathbb{G}$.

Our proof combines techniques from [38, 39, 6] with new ideas. Intuitively a signature in our scheme is a proof that either $\mathsf{DL}_{h_0}(z_0) = \mathsf{DL}_g(y_0)$ $(= x_0)$ or $\mathsf{DL}_{h_1}(z_1) = \mathsf{DL}_g(y_1)$ $(= x_1)$ (or both), or that a collision (including the discrete-log collision case) is found on one of the hash functions.

Proof (of Theorem 1). Our reduction is given a group $\mathbb{G}$ and a q_s-SDH challenge $\{g^{x^i}\}_{i=1,\ldots,q_s}$. Let us call $\mu_i = g^{x^i}$, for $i \in [1, q_s]$.

The reduction algorithm uses an existential forger $\mathcal{A}$ against our signature scheme to solve this challenge, *i.e.,* to find $g^{\frac{1}{x+\alpha}}$ for some α or to find collisions of one of the three above types. The reduction picks a random integer $\delta \in \{0, 1, 2\}$ and runs the subroutine Simulation δ described below.

$\boxed{\textbf{Simulation 0}}$

In this simulation, we simulate the Side 0 of the signature scheme while knowing the private key associated with Side 1, *i.e.,* the simulator knows x_1 but not x_0. The subroutine poses $y_0 = \mu_1$, randomly picks $x_1 \in \mathbb{Z}_q$ and sets $y_1 = g^{x_1}$. If $y_1 = y_0$, we know that $x_0 = x_1$ and so the q_s-SDH challenge can be solved easily. Therefore we assume that $x_0 \neq x_1$. A random generator $g_1 \in \mathbb{G}$ is generated as well.

Initialization: The simulator prepares q_s random tuples $(s_{0,i}, c_{0,i}, k_{1,i}) \in \mathbb{Z}_q^3$ and computes

$$\begin{aligned} u_{0,i} &= g^{s_{0,i}}\, {y_0}^{-c_{0,i}} \\ u_{1,i} &= g^{k_{1,i}} \\ \alpha_i &= \mathcal{H}(u_{0,i}, u_{1,i}). \end{aligned}$$

The simulator checks whether one of the α_i's is actually equal to $q - x_0$, in which (unlikely) case, the simulator sets $x = q - \alpha_i$ and directly solves the q_s-SDH problem. Similarly, it checks that $\alpha_i \neq q - x_1$, in which case initialization is restarted. Thus we now assume that $\alpha_i + x_0 \neq 0 \bmod q$ and $\alpha_i + x_1 \neq 0 \bmod q$, and so are invertible modulo q.

Let f be the polynomial $f(X) = \prod_{j=1}^{j=q_s}(X + \alpha_j)$. Let $g_0 = g^{\theta f(x)}$, for a random $\theta \in \mathbb{Z}_q$. By Lemma 1, g_0 can be simply computed thanks to μ_i's. Now the simulator runs $\mathcal{A}$ on the public key (g, g_0, g_1, y_0, y_1) and the public parameters $(q, \mathcal{G}, \mathcal{H}, \mathbb{G})$.

Simulating signature queries: Let $m_i \in \mathcal{M}$ be the i-th signature query. Simulator 0 behaves as follows. Using Lemma 1, $h_{0,i} = g_0{}^{\frac{1}{x_0+\alpha_i}}$ is easily computed, without unknown secret x_0. The simulator then computes $z_{0,i} = g_0\, h_{0,i}{}^{-\alpha_i}$ and $v_{0,i} = h_{0,i}{}^{s_{0,i}}\, z_{0,i}{}^{-c_{0,i}}$.
Now that all variables from Side 0 are simulated, the simulator uses x_1 to generate the variables from Side 1 as

$$\begin{aligned} h_{1,i} &= g_1{}^{\frac{1}{x_1+\alpha_i}} \\ v_{1,i} &= h_1{}^{k_{1,i}} \\ z_{1,i} &= h_1{}^{x_1} \end{aligned}$$

Finally, the simulator computes $d_i = \mathcal{G}(m_i, g, h_{0,i}, y_0, z_{0,i}, u_{0,i}, v_{0,i}, h_{1,i}, y_1, z_{1,i}, u_{1,i}, v_{1,i})$ and set $c_{1,i} = d_i - c_{0,i} \bmod q$. Finally, $s_{1,i} = k_{1,i} + c_{1,i} \cdot x_1 \bmod q$ is derived.
As one can see, signature $(h_{0,i}, s_{0,i}, c_{0,i}, h_{1,i}, s_{1,i}, c_{1,i})$ is valid and distributed as a regular signature (notably, $\mathsf{DL}_{h_0}(z_0) = \mathsf{DL}_g(y_0)$ and $\mathsf{DL}_{h_1}(z_1) = \mathsf{DL}_g(y_1)$).

Simulation 1

In this simulation, we proceed exactly as in Simulation 0, except that variables from the two sides are swapped: Side 1 is simulated using the μ_i's and Side 0 is trivial since the key x_0 is known by the simulation.

Simulation 2

The purpose of this simulation is not to solve the SDH instance, but to find a discrete-log collision on $\mathcal{G}$. Thus, in this simulation, the simulator knows the two keys x_0 and x_1, and responds to the adversary's signature queries as per the definition of the signing procedure.

Solving the q_s-SDH problem or finding collisions

This completes the description of our three simulation subroutines. Note that the three routines yield perfectly simulated distributions and are indistinguishable from one another from the adversary's perspective. We now focus on what our reduction does assuming that a forgery is returned by the forger at the end of the game.
Let $\sigma = (h_0, s_0, c_0, h_1, s_1, c_1) \in \mathbb{G}^{\diamond} \times \mathbb{Z}_q^2 \times \mathbb{G}^{\diamond} \times \mathbb{Z}_q^2$ be the valid forgery on a new message[12] $m \in \mathcal{M}$. Our reduction algorithm easily computes the other variables $u_0, v_0, z_0, u_1, v_1, z_1, \alpha, d$ by following the verification procedure. The reduction then checks whether $\alpha = \alpha_i$ for some $i = 1, ..., q_s$. Several cases and sub-cases appear.

[12] Hence, our scheme does not ensure strong existential unforgeability, but only existential unforgeability, which is sufficient in most usages.

■ **Case 1: $\boldsymbol{\alpha = \alpha_i}$.** This case can be subdivided into three sub-cases.

Case 1.1: $(u_0, u_1) \neq (u_{0,i}, u_{1,i})$. Then, (u_0, u_1) and $(u_{0,i}, u_{1,i})$ is a pair which forms a collision on $\mathcal{H}$ function. This probability is captured by $\varepsilon_{\mathcal{H}}$.

Case 1.2: $(u_0, u_1) = (u_{0,i}, u_{1,i})$ and $(c_0, c_1) \neq (c_{0,i}, c_{1,i})$. If $\delta \in \{0, 1\}$ and $c_\delta \neq c_{\delta,i}$, the reduction directly finds x and subsequently solves the q_s-SDH challenge: indeed, we have equations $u_\delta = u_{\delta,i}$, $u_{\delta,i} = g^{s_{\delta,i}}\, {y_\delta}^{-c_{\delta,i}}$ (from the signature queries) and $u_\delta = g^{s_\delta}\, {y_\delta}^{-c_\delta}$ (from the forge). Since δ is independent from the adversary, $\delta \in \{0, 1\}$ happens with a probability $\frac{2}{3}$ and $c_\delta \neq c_{\delta,i}$ knowing that $(c_0, c_1) \neq (c_{0,i}, c_{1,i})$ happens with a probability of $\frac{1}{2}$. This case probability is thus upper-bounded by the $3\,\varepsilon_{\texttt{SDH},1}$ term.

Case 1.3: $(u_0, u_1) = (u_{0,i}, u_{1,i})$ and $(c_0, c_1) = (c_{0,i}, c_{1,i})$. We know by the verification step that $d_i = c_{0,i} + c_{1,i} \bmod q$, and thus $d_i = d$. In other words,

$$(m, g, h_0, y_0, z_0, u_0, v_0, h_1, y_1, z_1, u_1, v_1)$$

and

$$(m_i, g, h_{0,i}, y_0, z_{0,i}, u_{0,i}, v_{0,i}, h_{1,i}, y_1, z_{1,i}, u_{1,i}, v_{1,i})$$

constitute a (classical) $\mathcal{G}$ collision.[13] This probability is captured by $\varepsilon_{\mathcal{G}}$.

Summing up, we get

$$\varepsilon_{\mathcal{H}} + 3\,\varepsilon_{\texttt{SDH},1} + \varepsilon_{\mathcal{G}} \geq \varepsilon \cdot \Pr[\texttt{Case 1}]$$

■ **Case 2: $\boldsymbol{\alpha \neq \alpha_i}$.** Since h_0 and h_1 are generators of $\mathbb{G}$ (this is notably the reason why we must check that they are not equal to 1 in the verification step), there exists a unique tuple $(k_0, k_0', k_0'', k_1, k_1', k_1'', x_0', x_1') \in \mathbb{Z}_q^2 \times \mathbb{Z}_q^\diamond \times \mathbb{Z}_q^2 \times \mathbb{Z}_q^\diamond \times \mathbb{Z}_q^2$ such that

$$\begin{aligned} u_0 &= g^{k_0}, & u_1 &= g^{k_1} \\ v_0 &= {h_0}^{k_0'}, & v_1 &= {h_1}^{k_1'} \\ z_0 &= {h_0}^{x_0'}, & z_1 &= {h_1}^{x_1'} \\ h_0 &= g^{k_0''}, & h_1 &= g^{k_1''}. \end{aligned}$$

Our goal is to prove that, with overwhelming probability, either ($k_0 = k_0'$ and $x_0 = x_0'$) or ($k_1 = k_1'$ and $x_1 = x_1'$), or both. This is somehow similar to EDL's proofs. By the verification step, we know that (all computations being modulo q)

$$\begin{aligned} k_0 &= s_0 - x_0 \cdot c_0, & k_1 &= s_1 - x_1 \cdot c_1 \\ k_0' &= s_0 - x_0' \cdot c_0, & k_1' &= s_1 - x_1' \cdot c_1 \end{aligned}$$

[13] Remind that $m \neq m_i$, since the forgery is assumed to be valid.

and that $c_0 + c_1 = \mathcal{G}(m, g, h_0, y_0, z_0, u_0, v_0, h_1, y_1, z_1, u_1, v_1)$.

Case 2.1: $x_0 \neq x'_0$ and $x_1 \neq x'_1$: if the Simulation 2 was not executed, the reduction aborts. Else, the reduction knows the two parts of the signing key x_0 and x_1, and can actually check that $z_0 \neq {h_0}^{x_0}$ and $z_1 \neq {h_1}^{x_1}$, *i.e.,* that $x_0 \neq x'_0$ and $x_1 \neq x'_1$. Then, $c_0 = \frac{k_0 - k'_0}{x'_0 - x_0} \bmod q$ and $c_1 = \frac{k_1 - k'_1}{x'_1 - x_1} \bmod q$. This implies (modulo q)

$$\begin{aligned}\frac{k_0 - k'_0}{x'_0 - x_0} + \frac{k_1 - k'_1}{x'_1 - x_1} = \mathcal{G}(m, g, h_0, &g^{x_0}, {h_0}^{x'_0}, g^{k_0}, {h_0}^{k'_0}, h_1, \\ &g^{x_1}, {h_1}^{x'_1}, g^{k_1}, {h_1}^{k'_1})\end{aligned}$$

or written differently

$$\begin{aligned}\frac{k_0 - k'_0}{x'_0 - x_0} + \frac{k_1 - k'_1}{x'_1 - x_1} = \mathcal{G}(m, g, g^{k''_0}, &g^{x_0}, g^{x'_0 k''_0}, g^{k_0}, g^{k'_0 k''_0}, g^{k''_1}, \\ &g^{x_1}, g^{x'_1 k''_1}, g^{k_1}, g^{k'_1 k''_1})\end{aligned}$$

or, in a more evocative form, with $k''_0 = a_1$ and $k''_1 = a_6$ not zero

$$\begin{array}{ll} k''_0 = a_1 & k''_1 = a_6 \\ x_0 = a_2 & x_1 = a_7 \\ x'_0 = a_3/a_1 & x'_1 = a_8/a_6 \\ k_0 = a_4 & k_1 = a_9 \\ k'_0 = a_5/a_1 & k'_1 = a_{10}/a_6 \end{array}$$

and finally

$$\frac{a_1 a_4 - a_5}{a_3 - a_1 a_2} + \frac{a_6 a_9 - a_{10}}{a_8 - a_6 a_7} = \mathcal{G}(m, g, g^{a_1}, g^{a_2}, g^{a_3}, g^{a_4}, g^{a_5}, g^{a_6}, g^{a_7}, g^{a_8}, g^{a_9}, g^{a_{10}})$$

This provides a discrete-log collision on $\mathcal{G}$ function, for the function $\mathcal{F}_0$ defined in Section 4.3, or, in other words, proves that $\mathcal{G}$ is not correlation-intractable.

The reduction cannot provide the a_i discrete logarithms, but this is not a problem, since the discrete-log collision can be checked without them, by giving away (x_0, x_1) and the equations followed by $(m, g, h_0, y_0, z_0, u_0, v_0, h_1, y_1, z_1, u_1, v_1)$. The probability of this event is counted by $3\,\varepsilon_{\mathsf{dl}_{\mathcal{G},\mathcal{F}_0}}$.

Case 2.2: $x_\delta = x'_\delta$. Then, $z_\delta = {h_\delta}^{x_\delta}$ and $h_\delta = {g_\delta}^{\frac{1}{x+\alpha}}$. As the simulation actually executed is unknown to the adversary, this happens with probability $\frac{1}{3}$. Thanks to Lemma 1, the reduction can return $(\alpha, g^{\frac{1}{x+\alpha}})$ as our answer to the q_s-SDH problem.

Summarizing, we have

$$3\,\varepsilon_{\mathsf{dl}_{\mathcal{G},\mathcal{F}_0}} + 3\,\varepsilon_{\mathtt{q-SDH},2} \geq \varepsilon \cdot \Pr[\mathtt{new}]$$

which proves the theorem.

5 Conclusion

In this paper, we have introduced a new signature scheme which is efficient and does not rely on pairings. Our scheme is provably secure under the strong Diffie-Hellman assumption and the correlation-intractability of the hash function. This scheme is our best attempt to extend the EDL family without the random oracle model.

An open question remains in how to reduce the size of our signatures and how to completely get rid of the assumption that finding discrete-log collisions for $\mathcal{G}$ function is intractable. More generally, one may try to extend EDL family in the standard model with other techniques, or to rely on weaker assumptions.

Acknowledgments. The author would like to thank the anonymous referees and Marc Fischlin for their useful remarks, Pascal Paillier and Marc Joye for their careful reading of this paper and finally Jeremy Bradley-Silverio Donato for his edits. More personal thanks go to Amal and Mathieu for their continous support.

References

1. Abe, M., Ambrona, M., Bogdanov, A., Ohkubo, M., Rose, A.: Non-interactive composition of sigma-protocols via share-then-hash. In: Advances in Cryptology - ASIACRYPT 2020. pp. 749–773. Springer (2020)
2. Barreto, P.S.L.M., Kim, H.Y., Lynn, B., Scott, M.: Efficient algorithms for pairing-based cryptosystems. In: Advances in Cryptology – CRYPTO 2002. Lecture Notes in Computer Science, vol. 2442, pp. 354–368. Springer-Verlag (2002)
3. Barreto, P.S.L.M., Naehrig, M.: Pairing-friendly elliptic curves of prime order. In: Selected Areas in Cryptography – SAC 2005. Lecture Notes in Computer Science, vol. 3897, pp. 319–331. Springer-Verlag (2005)
4. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: ACM Conference on Computer and Communications Security – ACM CCS 1993. pp. 62–73. ACM Press (1993)
5. Bellare, M., Rogaway, P.: The exact security of digital signatures - How to sign with RSA and Rabin. In: Advances in Cryptology – EUROCRYPT '96. Lecture Notes in Computer Science, vol. 1070, pp. 399–416. Springer-Verlag (1996)
6. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Advances in Cryptology – EUROCRYPT 2004. Lecture Notes in Computer Science, vol. 3027, pp. 56–73. Springer-Verlag (2004)
7. Boneh, D., Boyen, X.: Short signatures without random oracles and the SDH assumption in bilinear groups. Journal of Cryptology **21**(2), 149–177 (2008)
8. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. In: Advances in Cryptology – CRYPTO 2001. Lecture Notes in Computer Science, vol. 2139, pp. 213–229. Springer-Verlag (2001)
9. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. SIAM Journal on Computing **32**(3), 586–615 (2003)
10. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. Journal of Cryptology **17**(4), 297–319 (2004)

11. Camenisch, J., Lysyanskaya, A.: A signature scheme with efficient protocols. In: Security in Communication Networks – SCN 2002. Lecture Notes in Computer Science, vol. 2576, pp. 268–289. Springer-Verlag (2002)
12. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Advances in Cryptology – CRYPTO 2004. Lecture Notes in Computer Science, vol. 3152, pp. 56–72. Springer-Verlag (2004)
13. Canetti, R., Chen, Y., Reyzin, L., Rothblum, R.D.: Fiat-Shamir and correlation intractability from strong kdm-secure encryption. In: Advances in Cryptology - EUROCRYPT 2018. Lecture Notes in Computer Science, vol. 10820, pp. 91–122. Springer (2018)
14. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited (preliminary version). In: ACM Symposium on the Theory of Computing – STOC '98. pp. 209–218. ACM Press (1998)
15. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. Journal of the ACM **51**(4), 557–594 (2004)
16. Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Advances in Cryptology – CRYPTO '92. Lecture Notes in Computer Science, vol. 740, pp. 89–105. Springer-Verlag (1992)
17. Cheon, J.H.: Security analysis of the strong Diffie-Hellman problem. In: Advances in Cryptology – EUROCRYPT '2006. Lecture Notes in Computer Science, vol. 4004, pp. 1–11. Springer-Verlag (2006)
18. Chevallier-Mames, B.: An efficient CDH-based signature scheme with a tight security reduction. In: Advances in Cryptology – CRYPTO 2005. Lecture Notes in Computer Science, vol. 3621, pp. 511–526. Springer-Verlag (2005)
19. Chevallier-Mames, B., Joye, M.: A practical and tightly secure signature scheme without hash function. In: Topics in Cryptology – CT-RSA 2007. Lecture Notes in Computer Science, Springer-Verlag (2007)
20. Coron, J.S.: On the exact security of full domain hash. In: Advances in Cryptology – CRYPTO 2000. Lecture Notes in Computer Science, vol. 1880, pp. 229–235. Springer-Verlag (2000)
21. Couteau, G., Katsumata, S., Ursu, B.: Non-interactive zero-knowledge in pairing-free groups from weaker assumptions. In: Advances in Cryptology - EUROCRYPT 2020. Lecture Notes in Computer Science, vol. 12107, pp. 442–471. Springer (2020)
22. Cramer, R., Damgård, I.: Secure signature schemes based on interactive protocols. In: Advances in Cryptology - CRYPTO '95. Lecture Notes in Computer Science, vol. 963, pp. 297–310. Springer (1995)
23. Cramer, R., Damgård, I.: New generation of secure and practical RSA-based signatures. In: Advances in Cryptology – CRYPTO '96. Lecture Notes in Computer Science, vol. 1109, pp. 173–185. Springer-Verlag (1996)
24. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Advances in Cryptology - CRYPTO '94. Lecture Notes in Computer Science, vol. 839, pp. 174–187. Springer (1994)
25. Cramer, R., Shoup, V.: Signature schemes based on the strong RSA assumption. ACM Transactions on Information and System Security (TISSEC) **3**(3), 161–185 (2000)
26. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Transactions on Information Theory **22**(6), 644–654 (1976)
27. Dodis, Y., Oliveira, R., Pietrzak, K.: On the generic insecurity of the Full Domain Hash. In: Advances in Cryptology – CRYPTO 2005. Lecture Notes in Computer Science, vol. 3621, pp. 449–466. Springer-Verlag (2005)

28. Döttling, N., Garg, S.: Identity-based encryption from the Diffie-Hellman assumption. In: Advances in Cryptology - CRYPTO 2017. Lecture Notes in Computer Science, vol. 10401, pp. 537–569. Springer (2017)
29. Dwork, C., Naor, M.: An efficient existentially unforgeable signature scheme and its applications. In: Advances in Cryptology – CRYPTO '94. Lecture Notes in Computer Science, vol. 839, pp. 234–246. Springer-Verlag (1994)
30. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory **31**(4), 469–472 (1985)
31. Fiat, A., Shamir, A.: How to prove yourself: practical solutions to identification and signature problems. In: Advances in Cryptology – CRYPTO '86. Lecture Notes in Computer Science, vol. 263, pp. 186–184. Springer-Verlag (1986)
32. Fischlin, M.: The Cramer-Shoup strong-RSA signature scheme revisited. In: Public Key Cryptography – PKC 2003. Lecture Notes in Computer Science, vol. 2567, pp. 116–129. Springer-Verlag (2003)
33. Fischlin, M., Harasser, P., Janson, C.: Signatures from sequential-OR proofs. In: Advances in Cryptology - EUROCRYPT 2020. Lecture Notes in Computer Science, vol. 12107, pp. 212–244. Springer (2020)
34. Fischlin, M., Lehmann, A., Ristenpart, T., Shrimpton, T., Stam, M., Tessaro, S.: Random oracles with(out) programmability. In: Advances in Cryptology - ASIACRYPT 2010. Lecture Notes in Computer Science, vol. 6477, pp. 303–320. Springer-Verlag (2010)
35. Freeman, D., Scott, M., Teske, E.: A taxonomy of pairing-friendly elliptic curves. Journal of Cryptology **23**(2), 224–280 (2010)
36. Gennaro, R., Halevi, S., Rabin, T.: Secure hash-and-sign signatures without the random oracle. In: Advances in Cryptology – EUROCRYPT '99. Lecture Notes in Computer Science, vol. 1592, pp. 123–139. Springer-Verlag (1999)
37. Girault, M.: Self-certified public keys. In: Advances in Cryptology – EUROCRYPT '91. Lecture Notes in Computer Science, vol. 547, pp. 490–497. Springer-Verlag (1991)
38. Goh, E.J., Jarecki, S.: A signature scheme as secure as the Diffie-Hellman problem. In: Advances in Cryptology – EUROCRYPT 2003. Lecture Notes in Computer Science, vol. 2656, pp. 401–415. Springer-Verlag (2003)
39. Goh, E.J., Jarecki, S., Katz, J., Wang, N.: Efficient signature schemes with tight security reductions to the Diffie-Hellman problems. Journal of Cryptology (2007)
40. Goldwasser, S., Kalai, Y.T.: On the (in)security of the Fiat-Shamir paradigm. In: FOCS. pp. 102–113. IEEE Computer Society (2003)
41. Goldwasser, S., Micali, S., Rivest, R.L.: A "paradoxical" solution to the signature problem (extended abstract). In: Symposium on Foundations of Computer Science – FOCS '84. pp. 441–448. IEEE Press (1984)
42. Goldwasser, S., Micali, S., Rivest, R.L.: A digital signature scheme secure against adaptive chosen-message attacks. SIAM Journal on Computing **17**(2), 281–308 (1988)
43. Guillou, L.C., Quisquater, J.J.: A practical zero-knowledge protocol fitted to security microprocessor minimizing both trasmission and memory. In: Advances in Cryptology – EUROCRYPT '88. Lecture Notes in Computer Science, vol. 330, pp. 123–128. Springer-Verlag (1988)
44. Hess, F., Smart, N.P., Vercauteren, F.: The eta pairing revisited. IEEE Transactions on Information Theory **52**(10), 4595–4602 (2006)
45. Hofheinz, D., Kiltz, E.: Programmable hash functions and their applications. Journal of Cryptology **25**(3), 484–527 (2012)

46. Hohenberger, S., Waters, B.: Realizing hash-and-sign signatures under standard assumptions. In: Advances in Cryptology – EUROCRYPT '2009. Lecture Notes in Computer Science, vol. 5479, pp. 333–350. Springer-Verlag (2009)
47. Hohenberger, S., Waters, B.: Short and stateless signatures from the RSA assumption. In: Advances in Cryptology – CRYPTO 2009. Lecture Notes in Computer Science, vol. 5677, pp. 654–670. Springer-Verlag (2009)
48. Jakobsson, M., Schnorr, C.P.: Efficient oblivious proofs of correct exponentiation. In: Communications and Multimedia Security – CMS 1999. IFIP Conference Proceedings, vol. 152, pp. 71–86. IFIP (1999)
49. Joux, A.: A one round protocol for tripartite Diffie-Hellman. In: ANTS. Lecture Notes in Computer Science, vol. 1838, pp. 385–394. Springer-Verlag (2000)
50. Kalai, Y.T., Rothblum, G.N., Rothblum, R.D.: From obfuscation to the security of Fiat-Shamir for proofs. In: Advances in Cryptology - CRYPTO 2017. Lecture Notes in Computer Science, vol. 10402, pp. 224–251. Springer (2017)
51. Katz, J., Wang, N.: Efficiency improvements for signature schemes with tight security reductions. In: ACM Conference on Computer and Communications Security – ACM CCS 2003. pp. 155–164. ACM Press (2003)
52. Krawczyk, H., Rabin, T.: Chameleon signatures. In: Network and Distributed System Security Symposium – NDSS 2000. pp. 143–154 (2000)
53. Kurosawa, K., Schmidt-Samoa, K.: New online/offline signature schemes without random oracles. In: Public Key Cryptography – PKC 2006. Lecture Notes in Computer Science, vol. 3958, pp. 330–346. Springer-Verlag (2006)
54. Micali, S., Reyzin, L.: Improving the exact security of digital signature schemes. Journal of Cryptology **15**(1), 1–18 (2002)
55. Naccache, D., Pointcheval, D., Stern, J.: Twin signatures: an alternative to the hash-and-sign paradigm. In: ACM Conference on Computer and Communications Security – ACM CCS 2001. pp. 20–27. ACM Press (2001)
56. Naor, M., Yung, M.: Universal one-way hash functions and their cryptographic applications. In: Proceedings of the 21st Annual ACM Symposium on Theory of Computing, 1989. pp. 33–43. ACM (1989)
57. Paillier, P.: Impossibility proofs for RSA signatures in the standard model. In: Topics in Cryptology – CT-RSA 2007. Lecture Notes in Computer Science, vol. 4377, pp. 31–48. Springer-Verlag (2007)
58. Paillier, P., Vergnaud, D.: Discrete-log-based signatures may not be equivalent to discrete log. In: Advances in Cryptology – ASIACRYPT 2005. Lecture Notes in Computer Science, vol. 3788, pp. 1–20. Springer-Verlag (2005)
59. Pointcheval, D., Stern, J.: Security proofs for signature schemes. In: Advances in Cryptology – EUROCRYPT '96. Lecture Notes in Computer Science, vol. 1070, pp. 387–398. Springer-Verlag (1996)
60. Poupard, G., Stern, J.: Security analysis of a practical "on the fly" authentication and signature generation. In: Advances in Cryptology – EUROCRYPT '98. Lecture Notes in Computer Science, vol. 1403, pp. 422–436. Springer-Verlag (1998)
61. Rabin, M.O.: Digital signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science (Janvier 1979)
62. Rivest, R.L., Shamir, A., Adleman, L.M.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM **21**(2), 120–126 (1978)
63. Rompel, J.: One-way functions are necessary and sufficient for secure signatures. In: Proceedings of the 22nd Annual ACM Symposium on Theory of Computing 1990. pp. 387–394. ACM (1990)

64. Schnorr, C.P.: Efficient signature generation by smart cards. Journal of Cryptology **4**(3), 161–174 (1991)
65. Seurin, Y.: On the exact security of Schnorr-type signatures in the random oracle model. In: Advances in Cryptology - EUROCRYPT 2012. Lecture Notes in Computer Science, vol. 7237, pp. 554–571. Springer-Verlag (2012)
66. Shamir, A., Tauman, Y.: Improved online/offline signature schemes. In: Advances in Cryptology – CRYPTO 2001. Lecture Notes in Computer Science, vol. 2139, pp. 355–367. Springer-Verlag (2001)
67. Stern, J.: Why provable security matters? In: Advances in Cryptology – EUROCRYPT 2003. Lecture Notes in Computer Science, vol. 2656, pp. 449–461. Springer-Verlag (2003)
68. Stern, J., Pointcheval, D., Malone-Lee, J., Smart, N.P.: Flaws in applying proof methodologies to signature schemes. In: Advances in Cryptology – CRYPTO 2002. Lecture Notes in Computer Science, vol. 2442, pp. 93–110. Springer-Verlag (2002)
69. Waters, B.: Efficient identity-based encryption without random oracles. In: Advances in Cryptology – EUROCRYPT 2005. Lecture Notes in Computer Science, vol. 3494, pp. 114–127. Springer-Verlag (2005)
70. Zhang, F., Safavi-Naini, R., Susilo, W.: An efficient signature scheme from bilinear pairings and its applications. In: Public Key Cryptography – PKC 2004. Lecture Notes in Computer Science, vol. 2947. Springer-Verlag (2004)