Universal Designated Verifier Signatures Without Random Oracles or Non-Black Box Assumptions

Fabien Laguillaumie[1], Benoît Libert[2⋆], and Jean-Jacques Quisquater[2]

[1] Projet TANC - INRIA Futurs
Laboratoire d'informatique (LIX)
École polytechnique, 91128 Palaiseau cedex, France
[2] UCL Crypto Group
Place du Levant, 3. B-1348 Louvain-La-Neuve, Belgium

Abstract. Universal designated verifier signatures (UDVS) were introduced in 2003 by Steinfeld *et al.* to allow signature holders to monitor the verification of a given signature in the sense that any plain signature can be publicly turned into a signature which is only verifiable by some specific designated verifier. Privacy issues, like non-dissemination of digital certificates, are the main motivations to study such primitives. In this paper, we propose two fairly efficient UDVS schemes which are secure (in terms of unforgeability and anonymity) in the *standard model* (i.e. without random oracles). Their security relies on algorithmic assumptions which are much more *classical* than assumptions involved in the two only known UDVS schemes in standard model to date. The latter schemes, put forth by Zhang *et al.* in 2005 and Vergnaud in 2006, rely on the Strong Diffie-Hellman assumption and the strange-looking *knowledge of exponent assumption* (KEA). Our schemes are obtained from Waters's signature and they do not need the KEA assumption. They are also the first random oracle-free constructions with the anonymity property.

1 Introduction

Many electronic applications have a crucial need for privacy which has been of central interest in the cryptographic community since the early eighties with the introduction of "special-purpose signatures". In 2003, Steinfeld *et al.* [31] suggested the idea of transforming a digital signature into a certain special signature (designated verifier). This notion is very useful in the design of sensitive e-applications with privacy issues. In this paper, we make a step forward in this area by designing efficient schemes which are secure without random oracles under more classical assumptions than the previous secure schemes that are secure in a standard model of computation.

Designated Verifier Signatures. Designated verifier proofs were introduced in 1996 by Jakobsson, Sako and Impagliazzo [20] in order to serve during confirmation and denial procedures of undeniable signatures [12] with the motivation

⋆ This author thanks the DGTRE's First Europe Program of the Walloon Region in Belgium.

to face *blackmailing* or *mafia* attacks. Designated verifier signatures (DVS) were built using designated verifier proofs to convince a unique verifier chosen by the signer so that the verifier cannot transfer his conviction regarding the correctness of the signature. Roughly speaking, DVS schemes were obtained from Jakobsson *et al.*'s designated verifier proofs via the Fiat-Shamir heuristic [16].

Several years after the seminal paper of Jakobsson *et al.* [20], many new schemes appeared in the literature, but not always with a precise formalization of security requirements. The first "modern" scheme, proposed by Saeednia *et al.* in 2003 [29], was based on Schnorr's signature [30] but it still was not supported by a formal security model. At Asiacrypt'03, Steinfeld, Bull, Wang and Pieprzyk [31] gave a new proper definition of unforgeability. Laguillaumie and Vergnaud [23] subsequently adapted the notion of anonymity for undeniable signatures to the context of DVS schemes: they defined the *privacy signer's identity*, which protects the anonymity of the signer and captures the notion of *strong* DVS introduced in [20]. The notion of DVS schemes was then extended in [24] to allow the designation of several verifiers. More recently, Bender, Katz and Morselli [6] described 2-user ring signatures that immediately give rise to designated verifier signatures in the standard model.

Universal Designated Verifier Signature. Along with a formal security model for DVS schemes, Steinfeld *et al.* [31] defined a new useful property for traditional signatures. Basically, anyone holding a valid digital signature should be able to transform it so that only a specific user is able to ascertain the correctness of the signature. This transformation removes the self-authenticating property of signatures, and the resulting process was called *universal designated verifier signatures* (UDVS). At PKC'04, Steinfeld, Wang and Pierpzyk [32] proposed UDVS extensions of Schnorr and RSA signatures. A further extension termed "universal multi-designated verifier signatures" was considered in [27].

Except [6], all aforementioned works conduct security analyzes in the random oracle model [5] where hash functions are viewed as idealized random functions. As security in this model does *not* [10] imply the security in the real world, an important effort is currently achieved to avoid it and obtain security results in the standard model. A pairing-based random oracle-free signature algorithm due to Boneh and Boyen [7] is often used in the design of special-purpose signatures such as UDVS schemes put forth by Zhang *et al.* [35] and Vergnaud [33]. Nonetheless, the computational assumption (called Strong Diffie-Hellman assumption or SDH for short) underlying the Boneh-Boyen scheme is ad-hoc and non-standard. Besides, security proofs of schemes in [33,35] additionally need an even stronger and odd assumption known as the *knowledge-of-exponent assumption*[1] (KEA) [4,14,19] which is *non-black box* in that security reductions from

[1] Intuitively, this assumption states that, given $(g, h = g^a)$ in a cyclic group $\mathbb{G} = \langle g \rangle$, the only way to generate pairs $(y_1, y_2) \in \mathbb{G} \times \mathbb{G}$ s.t. $y_2 = y_1^a$ without knowing a is to set $y_1 = g^r$ and $y_2 = h^r$ for a randomly chosen r. Any adversary $\mathcal{A}$ producing such a pair (y_1, y_2) necessarily "knows" the exponent r that could be extracted by accessing $\mathcal{A}$'s memory.

this assumption entail some kind of access to the internal state of the adversary.

Our contributions. Avoiding random oracles in security proofs often leads to use strong and ad-hoc assumptions. Hence, actual security benefits of this breakthrough are not always clear. The only secure [33, 35] UDVS schemes in the standard model rely on the combined SDH and KEA assumptions. The former was recently reconsidered [13] and the latter is non-black box and so odd that it is generally disliked and avoided whenever possible although it holds in generic groups [15]. It was shown in [33] that the KEA may be avoided in [33, 35], but both constructions then have a security resting on a very exotic assumption.

In this work, we aim at obtaining UDVS schemes satisfying strong security notions in the standard model and under more classical assumptions. We start from Waters's signature [34] which is (not strongly) existentially unforgeable under the Diffie-Hellman assumption in groups equipped with bilinear maps. We turn it into a UDVS scheme which is unforgeable under an alleviated version of the Gap Bilinear Diffie-Hellman assumption and protects the anonymity of signers under the Decisional Bilinear Diffie-Hellman assumption.

In a second step, we use a technique due to Boneh, Shen and Waters [9] to make our scheme *strongly* unforgeable. The main motivation to consider such an enhanced unforgeability is two-fold. First, it allows for a security resting on the weaker Bilinear Diffie-Hellman assumption (in other words, we bypass the use of a fancy decision oracle in the proof) at the expense of a loss of "tightness" in the reduction. Yet, the security of this variant relies on an assumption whose strength is totally independent of the number of adversarial queries (unlike [33, 35]). Underlying assumptions aside, our second scheme features a provable anonymity in a stronger sense (i.e. in a game where verification queries are allowed to adversaries). Our constructions turn out to be the *only* random oracle-free UDVS that meet an anonymity property in the "find-then-guess" sense following [23]. Indeed, solutions given in [33, 35] are provably not anonymous in this sense.

2 Ingredients

2.1 Universal Designated Verifier Signatures

Definition 1 (UDVS schemes). *A* universal designated verifier signature scheme *UDVS is a 5-tuple $\mathsf{UDVS} = (\Sigma, \mathsf{Register}, \mathsf{VKeyGen}, \mathsf{Designate}, \mathsf{DVerify})$ of algorithms parameterized by a security parameter k.*

- *$\Sigma = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ is a traditional digital signature scheme;*
- *$\mathsf{UDVS.Register}$ is a protocol between a "key registration authority" (KRA) and a user, both taking as input public parameters and the verifier's public key pk_v. The outcome is a notification decision from the KRA[2];*
- *$\mathsf{UDVS.VKeyGen}$ is a probabilistic algorithm which takes public parameters as input, and produces a pair of keys (sk_V, pk_V) for the designated verifier;*

[2] The protocol typically consists in having the KRA check that users know their private key.

– *$\mathsf{UDVS.Designate}$ is a (possibly probabilistic) algorithm which takes as inputs public parameters, a public key pk_S, a message m, a putative signature σ on m with respect to the public key pk_S, and the public key of a designated verifier pk_V, and produces a designated verifier signature $\tilde{\sigma}$;*
– *$\mathsf{UDVS.DVerify}$ is a deterministic algorithm which takes as inputs public parameters, a message m, a putative designated verifier signature $\tilde{\sigma}$, a public key pk_S, a pair of keys (sk_V, pk_V). The output is 1 if the signature $\tilde{\sigma}$ is accepted and 0 otherwise.*

The usual correctness requirement imposes that correctly formed plain or designated signatures are always accepted by the relevant verification algorithm.

In terms of security, an UDVS scheme must fit a natural variant of the standard notion of *existential unforgeability under chosen-message attacks* [18]. It should also achieve two anonymity properties: (1) the notion of (unconditional) *source hiding* which is the ambiguity about whom among the signer and the designated verifier a signature emanates from; (2) the *signer's privacy*, which is analogous to the notion of anonymity for undeniable signatures.

Source hiding. An UDVS scheme is *source hiding* if there exists an algorithm that takes as input only the secret key of the designated verifier and which produces bit strings which are perfectly indistinguishable (even knowing all secret keys) from the distribution of actual designated verifier signatures.

Unforgeability. We consider the notion of unforgeability introduced in [32] which is an extension of the *chosen-message* security introduced in [18]. Informally speaking, an attacker is given a signer's public key pk_S, a designated verifier's public key pk_V and access to a signing oracle and a verification oracle. He should be unable to produce a signature on a new message.

Definition 2. *An UDVS scheme is said (not strongly) existentially unforgeable if no PPT adversary $\mathcal{F}$ has a non-negligible advantage in the following game.*

1. *The challenger $\mathcal{C}$ takes as input a security parameter k and executes $\mathsf{params} \leftarrow \mathsf{UDVS}.\Sigma.\mathsf{Setup}(k)$, $(sk_S^\star, pk_S^\star) \leftarrow \mathsf{UDVS}.\Sigma.\mathsf{KeyGen}(k, \mathsf{params})$, $(sk_V^\star, pk_V^\star) \leftarrow \mathsf{UDVS.VKeyGen}(k, \mathsf{params})$. It gives $pk_S^\star$ and $pk_V^\star$ to the forger $\mathcal{F}$ and keeps $sk_S^\star$ and $sk_V^\star$ to itself.*
2. *The forger $\mathcal{F}$ can issue the following queries:*
   i) a registration query for a public key pk; the attacker engages in the registration protocol with the KRA;
   ii) a signing query for some message m; the challenger $\mathcal{C}$ executes $\sigma \leftarrow \mathsf{UDVS}.\Sigma.\mathsf{Sign}(k, \mathsf{params}, m, sk_S^\star)$ and hands σ to $\mathcal{F}$;
   iii) a verification query for pairs $(m, \tilde{\sigma})$ of his choice; $\mathcal{C}$ returns to $\mathcal{F}$ the value $\mathsf{UDVS.DVerify}(k, \mathsf{params}, m, \tilde{\sigma}, pk, (sk_V^\star, pk_V^\star))$;
3. *$\mathcal{F}$ outputs a V-designated verifier signature $\tilde{\sigma}^\star$ for a* new *message $m^\star$.*

The adversary $\mathcal{F}$ succeeds if $\mathsf{UDVS.DVerify}\big(k, \mathsf{params}, pk_S^\star, (sk_V^\star, pk_V^\star)\big) = 1$ and $m^\star$ has not been asked by $\mathcal{F}$ in a signing query in step 2 of the game. An attacker $\mathcal{F}$ is said to $(\tau, q_s, q_v, \varepsilon)$-break the unforgeability of the UDVS scheme if he succeeds in the game within running time τ and with probability ε after having made q_s signing queries and q_v verification queries.

Strong Unforgeability. Definition 2 only captures the standard level of unforgeability. In the strengthened notion of *strong* unforgeability [1], the forger is allowed to output a fake designated signature $\tilde{\sigma}$ on a previously signed message $m^\star$. Here, we impose that $\tilde{\sigma}$ must differ from designated signatures obtained by applying the (deterministic) designation algorithm to all outputs of signing queries with input $m^\star$ during the game. We emphasize that this model only makes sense for schemes using a deterministic designation algorithm[3].

Privacy of signer's identity/Anonymity. Privacy of signer's identity was formally defined for designated verifier signatures by Laguillaumie and Vergnaud [23]. It captures the *strong anonymity* property introduced by Jakobsson *et al.* in [20]. Although designated verifier signatures are *signer ambiguous* regarding the signer and the designated verifier, it might remain possible to distinguish the actual issuer of a given signature between two potential signers. The next definition captures that it should be (computationally) infeasible. It is analogous to the notion of anonymity for undeniable signatures [17].

Definition 3. *An UDVS has the signer-privacy property if no PPT distinguisher $\mathcal{D}$ has a non-negligible advantage in the next game.*

1. *The challenger $\mathcal{C}$ takes as input a security parameter k and executes $\mathsf{params} \leftarrow \mathsf{UDVS}.\Sigma.\mathsf{Setup}(k)$, $(sk^\star_{S,0}, pk^\star_{S,0}), (sk^\star_{S,1}, pk^\star_{S,1}) \leftarrow \mathsf{UDVS}.\Sigma.\mathsf{KeyGen}(k, \mathsf{params})$, $(sk^\star_V, pk^\star_V) \leftarrow \mathsf{UDVS.VKeyGen}(k, \mathsf{params})$. It hands public keys $pk^\star_{S,0}, pk^\star_{S,1}$ and $pk^\star_V$ to $\mathcal{D}$ and keeps $sk^\star_{S,0}, sk^\star_{S,1}, sk^\star_V$ to itself.*
2. *The distinguisher $\mathcal{D}$ issues a number of queries exactly as in the game modeling the unforgeability property. Those queries may pertain to both of the challenge public keys $pk^\star_{S,0}, pk^\star_{S,1}$.*
3. *$\mathcal{D}$ produces a message $m^\star$ of her choosing. The challenger $\mathcal{C}$ then flips a fair coin $b^\star \overset{R}{\leftarrow} \{0,1\}$, generates a signature in the name of one of the signers $\sigma \leftarrow \mathsf{UDVS}.\Sigma.\mathsf{Sign}(k, \mathsf{params}, m^\star, sk^\star_{S,b^\star})$ and designates it into $\tilde{\sigma}^\star \leftarrow \mathsf{UDVS.Designate}(k, \mathsf{params}, pk^\star_{S,b^\star}, m, \sigma, pk^\star_V)$ which is sent to $\mathcal{D}$.*
4. *$\mathcal{D}$ issues new queries with the restriction of not querying $\tilde{\sigma}^\star$ for verification.*
5. *Eventually, $\mathcal{D}$ outputs a bit b and wins if $b = b^\star$.*

If $\mathcal{D}$ has advantage $\varepsilon = |Pr[b = b^\star] - 1/2|$ when making at most q_s and q_v signing and verification queries within running time τ, then we say that he $(\tau, q_s, q_v, \varepsilon)$-breaks the anonymity of the UDVS scheme.

[3] Defining strong unforgeability for schemes with probabilistic designation is more subtle. A reasonable option is the following. We still forbid *plain* signature queries for the message $m^\star$. Instead, $\mathcal{F}$ is equipped with a designated signing oracle taking as input a message m and some registered verifier's public key pk_B. The latter may differ from the target verifier's public key $pk^\star_V$ as long as it was registered and $\mathcal{F}$ proved her knowledge of the matching secret sk_B. The designated signing oracle first generates a plain signature $\sigma \leftarrow \mathsf{UDVS}.\Sigma.\mathsf{Sign}(k, \mathsf{params}, m, sk^\star_S)$ and designates it into $\tilde{\sigma} \leftarrow \mathsf{UDVS.Designate}(k, \mathsf{params}, pk, m, \sigma, pk_B)$ which is given to $\mathcal{F}$. The latter has to come up with a pair $(m^\star, \tilde{\sigma}^\star)$ designated to the target verifier $pk^\star_V$ and $(m^\star, \tilde{\sigma}^\star)$ may not result from a designated signing query with $pk^\star_V$ as a verifier's public key.

2.2 Bilinear Maps

We now recall basics about bilinear maps which are the main algebraic tool to design our new UDVS construction.

Definition 4. *Let* $(\mathbb{G}, +)$ *and* $(\mathbb{H}, \cdot)$ *be groups of prime order* q *and* $P \in \mathbb{G}$*. A* symmetric admissible bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{H}$ *has the following properties:*

1. *bilinearity:* $e(aP, bQ) = e(P, Q)^{ab}$ *for any* $(P, Q) \in \mathbb{G} \times \mathbb{G}$ *and* $a, b \in \mathbb{Z}$*;*
2. *efficient computability for any possible input pair;*
3. *non-degeneracy:* $e(P, P)$ *generates* $\mathbb{H}$ *whenever* P *generates* $\mathbb{G}$*.*

Definition 5. *A* BDH-parameter-generator *is a probabilistic algorithm that takes a security parameter* λ *as input and outputs a 5-tuple* $(q, P, \mathbb{G}, \mathbb{H}, e)$ *where* q *is a* λ*-bit prime number,* $(\mathbb{G}, +)$ *and* $(\mathbb{H}, \cdot)$ *are groups of order* q*,* $P \in \mathbb{G}$ *is a generator, and* $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{H}$ *is an admissible bilinear map.*

Complexity assumptions. Let $(q, P, \mathbb{G}, \mathbb{H}, e)$ be the output of a prime-order-BDH-parameter-generator for a security parameter k. Basically,

1. the (computational) **Bilinear Diffie-Hellman Problem** (BDH) [8, 21] is to compute $e(P, P)^{abc} \in \mathbb{H}$ given $(P, aP, bP, cP) \in \mathbb{G}^4$;
2. the **Decisional Bilinear Diffie-Hellman Problem** (DBDH) is to distinguish the distribution of BDH tuples $(aP, bP, cP, e(P, P)^{abc})$ from the distribution of random tuples $(aP, bP, cP, e(P, P)^z)$. We say that an algorithm $\mathcal{B}$ solving the DBDH problem has advantage ε if

$$\Big|\Pr[\mathcal{B}(P, aP, bP, cP, e(P, P)^{abc}) = 1 | a, b, c \overset{R}{\leftarrow} \mathbb{Z}_q^*] \\ - \Pr[\mathcal{B}(P, aP, bP, cP, e(P, P)^{z}) = 1 | a, b, c, z \overset{R}{\leftarrow} \mathbb{Z}_q^*]\Big| \geq \varepsilon;$$

3. the **Gap Bilinear Diffie-Hellman Problem** (GBDH) consists in solving the BDH problem (P, aP, bP, cP) with the help of an oracle deciding whether tuples $(P, xP, yP, zP, h) \in \mathbb{G}^4 \times \mathbb{H}$ satisfy $h = e(P, P)^{xyz}$;
4. the **weak Gap Bilinear Diffie-Hellman Problem** (wGDBH) is to solve a BDH instance $(P, aP, bP, cP) \in \mathbb{G}^4$ using a *restricted* decision oracle deciding whether pairs $(zP, h) \in \mathbb{G} \times \mathbb{H}$ satisfy $h = e(P, P)^{abz}$.

The last problem is not easier than the GBDH problem in that fewer degrees of freedom are allowed when using the decision oracle. We call *weak* Gap Bilinear Diffie-Hellman assumption its intractability for any PPT algorithm.

The security of our scheme relies on the wGBDH assumption which, although non-standard, is a black box assumption (see [28] for the historical definition of a *gap problem*). In section 5, we shall explain how to get rid of interactive assumptions and modify our scheme to end up with a security resting on the softer Bilinear Diffie-Hellman assumption.

3 Our UDVS Scheme

We present in this section the design of our new universal designated verifier signatures. It is based on Waters' signature scheme [34].

In our notation, hashed messages $\mathbf{m}$ are always represented as n-bit vectors $(m_1, \ldots, m_n)$ with $m_i \in \{0,1\}$ for all $i \in \{1, \ldots, n\}$.

- $\mathsf{UDVS}.\Sigma.\mathsf{Setup}$: public parameters include the output $(q, P, \mathbb{G}, \mathbb{H}, e)$ of a BDH-parameter-generator as well as an integer n, a collision-resistant hash function $h : \{0,1\}^* \to \{0,1\}^n$, random elements $P', U' \in \mathbb{G}$ and a random n-tuple $(U_1, \ldots, U_n) \in \mathbb{G}^n$. We call $F : \{0,1\}^n \to \mathbb{G}$ the application mapping strings $\mathbf{m}$ onto $F(\mathbf{m}) = U' + \sum_{i=1}^{n} m_i U_i$.
$$\mathsf{params} := \{n, q, \mathbb{G}, \mathbb{H}, e, P, P', U', U_1, \ldots, U_n, F, h\}.$$
- $\mathsf{UDVS}.\Sigma.\mathsf{KeyGen}$: a signer's private key is a randomly chosen $\alpha_S \overset{R}{\leftarrow} \mathbb{Z}_q^*$; his public key consists of a group element $P_S = \alpha_S P$.
- $\mathsf{UDVS}.\Sigma.\mathsf{Sign}$: given a message $M \in \{0,1\}^*$, the signer computes $\mathbf{m} = h(M)$ and picks $r \overset{R}{\leftarrow} \mathbb{Z}_q^*$. The signature is $\sigma = (\sigma_1, \sigma_2) = (\alpha_S P' + rF(\mathbf{m}), rP)$.
- $\mathsf{UDVS}.\mathsf{Register}$: a public key is registered by letting the user prove the knowledge of its secret key to the KRA.
- $\mathsf{UDVS}.\Sigma.\mathsf{Verify}$: a plain signature $\sigma = (\sigma_1, \sigma_2)$ on M is accepted if $e(\sigma_1, P) = e(P_S, P')e(\sigma_2, F(\mathbf{m}))$ where $\mathbf{m} = h(M)$.
- $\mathsf{UDVS}.\mathsf{VKeyGen}$: a designated verifier's private key is a random element $\alpha_V \overset{R}{\leftarrow} \mathbb{Z}_q^*$; the matching public key is $P_V = \alpha_V P \in \mathbb{G}$.
- $\mathsf{UDVS}.\mathsf{Designate}$: the holder of a signature $\sigma = (\sigma_1, \sigma_2)$, who chooses V as designated verifier produces the designated verifier signature $\tilde{\sigma} = (\tilde{\sigma}_1, \sigma_2)$ with $\tilde{\sigma_1} = e(\sigma_1, P_V)$.
- $\mathsf{UDVS}.\mathsf{DVerify}$: given a purported signature $(\tilde{\sigma}_1, \sigma_2)$, the designated verifier checks whether $\tilde{\sigma_1} = e(P_S, P')^{\alpha_V} e(\sigma_2, F(\mathbf{m}))^{\alpha_V}$ where $\mathbf{m} = h(M)$.

4 Security

Correctness and unconditional source hiding are straightforward.

4.1 Unforgeability

The proof of the next theorem follows the same strategy as the security proof of Waters's identity based encryption scheme [34].

Theorem 1. *Assuming that a forger $\mathcal{F}$ is able to $(t, q_s, q_v, \varepsilon)$-break the scheme, there is an algorithm $\mathcal{B}$ that (t', ε')-breaks the wGBDH assumption where*

$$\varepsilon' \geq \frac{\varepsilon}{4q_s(n+1)} \qquad t' \leq t + O((q_s + q_v)\tau_m + q_v\tau_p)$$

and τ_m, τ_p respectively denote the cost of a scalar multiplication in $\mathbb{G}$ and the time complexity of a pairing calculation.

Proof. Algorithm $\mathcal{B}$ is given a group $\mathbb{G}$ together with a generator P, elements $(aP, bP, cP) \in \mathbb{G}^3$ and an oracle $\mathcal{O}_{DBDH}(., bP, cP, .)$ deciding whether tuples of the shape $(aP, bP, cP, h) \in \mathbb{G}^3 \times H$ satisfy $h = e(P,P)^{abc}$. It uses $\mathcal{F}$ to extract $e(P,P)^{abc}$. The attack environment is simulated as follows.

Setup and key generation: $\mathcal{B}$ randomly chooses $k \in \{0, \ldots, n\}$ and defines $\ell = 2q_s$. We assume[4] that $\ell(n+1) < q$ which implies $0 \leq k\ell < q$. The simulator $\mathcal{B}$ randomly selects $x' \overset{R}{\leftarrow} \mathbb{Z}_\ell$ and a vector $(x_1, \ldots, x_n)$ of elements with $x_i \in \mathbb{Z}_\ell$ for all i. It also chooses at random an integer $y' \overset{R}{\leftarrow} \mathbb{Z}_q$ and a vector $(y_1, \ldots, y_n)$ with $y_j \in \mathbb{Z}_q$ for all j. For ease of explanation, we shall consider two functions

$$J(\mathbf{m}) = x' + \sum_{i=1}^{n} m_i x_i - k\ell \quad \text{and} \quad K(\mathbf{m}) = y' + \sum_{i=1}^{n} m_i y_i.$$

System-wide parameters are then chosen as $P' = cP$ and

$$U' = (x' - k\ell)P' + y'P \qquad U_i = x_i P' + y_i P \text{ for } 1 \leq i \leq n$$

which means that, for any string $\mathbf{m} \in \{0,1\}^n$, we have

$$F(\mathbf{m}) = U' + \sum_{i=1}^{n} m_i U_i = J(\mathbf{m})P' + K(\mathbf{m})P.$$

Besides, signer and verifier's public keys are set to $P_S = aP$ and $P_V = bP$.

Queries: once $\mathcal{F}$ is started with public parameters and public keys P_S, P_V as input, two kinds of queries may occur.

Signing queries: let $\mathbf{m} = h(M)$ be a message for which $\mathcal{F}$ requests a signature. If $J(\mathbf{m}) = 0 \bmod q$, $\mathcal{B}$ aborts. Otherwise, it can construct a signature by picking $r \overset{R}{\leftarrow} Z_q$ and computing

$$\sigma = (\sigma_1, \sigma_2) = \left(-\frac{K(\mathbf{m})}{J(\mathbf{m})} P_S + rF(\mathbf{m}), -\frac{1}{J(\mathbf{m})} P_S + rP \right).$$

If we define $\tilde{r} = r - a/J(\mathbf{m})$, σ is a valid signature as

$$\begin{aligned} \sigma_1 &= -\frac{K(\mathbf{m})}{J(\mathbf{m})} P_S + rF(\mathbf{m}) \\ &= -\frac{K(\mathbf{m})}{J(\mathbf{m})} P_S + \tilde{r}F(\mathbf{m}) + \frac{a}{J(\mathbf{m})}(J(\mathbf{m})P' + K(\mathbf{m})P) \\ &= aP' + \tilde{r}F(\mathbf{m}) \end{aligned}$$

and $\sigma_2 = (r - a/J(\mathbf{m}))P = \tilde{r}P$. The plain signature σ is then transformed using the public designation algorithm.

[4] This is a realistic requirement as parameters should be chosen s.t. $n \geq 160$, $q > 2^{160}$ and it is common to suppose $q_s < 2^{30}$.

Verification queries: at any time, $\mathcal{F}$ may enquire for the (in)validity of a designated signature $\tilde{\sigma} = (\tilde{\sigma_1}, \sigma_2)$ on a message $\mathbf{m} = h(M)$ and expects $\mathcal{B}$ to (in)validate it using the (unknown) private key $\alpha_V = b$. To answer such a query, $\mathcal{B}$ evaluates $J(\mathbf{m})$ and $K(\mathbf{m})$, invokes the decision oracle on the tuple $(P_S + J(\mathbf{m})\sigma_2, P_V, P', \tilde{\sigma_1}/e(K(\mathbf{m})\sigma_2, P_V))$ and returns 1 (meaning that $\tilde{\sigma}$ is a valid designated signature) if $\mathcal{O}_{DBDH}(.)$ deems it as a valid tuple. Otherwise, it returns 0 and declares $\tilde{\sigma}$ as invalid. We observe that, whenever $\tilde{\sigma}$ is correct, we have $\sigma_2 = rP$ and

$$\begin{aligned}\tilde{\sigma_1} &= e\left(aP' + r(J(\mathbf{m})P' + K(\mathbf{m})P), bP\right)\\ &= e\left(aP' + rJ(\mathbf{m})P', bP\right) e\left(K(\mathbf{m})rP, bP\right)\\ &= e\left(acP + J(\mathbf{m})rcP, bP\right) e\left(K(\mathbf{m})rP, bP\right)\\ &= e(P,P)^{(a+J(\mathbf{m})r)bc} e\left(K(\mathbf{m})\sigma_2, bP\right)\end{aligned}$$

for some $r \in \mathbb{Z}_q^*$ and $\tilde{\sigma_1}/e(K(\mathbf{m})\sigma_2, P_V)$ is the solution of the bilinear Diffie-Hellman instance

$$((a + rJ(\mathbf{m}))P, bP, cP) = (P_S + J(\mathbf{m})\sigma_2, P_V, P')\,.$$

If $\mathcal{F}$ ever issues such a verification query where $J(\mathbf{m}) = 0$ and $\mathcal{O}_{DBDH}(.)$ returns 1, $\mathcal{B}$ immediately halts and outputs $\tilde{\sigma_1}/e(K(\mathbf{m})\sigma_2, P_V)$.

Forgery: if $\mathcal{B}$ did not abort, $\mathcal{F}$ is expected to come with a fake designated signature $\tilde{\sigma}^\star = (\tilde{\sigma_1}^\star, \sigma_2)$ on some new message $\mathbf{m}^\star = h(M^\star)$. At that point, $\mathcal{B}$ reports "failure" if $J(\mathbf{m}^\star) \neq 0 \bmod q$. Otherwise, $F(\mathbf{m}^\star) = K(\mathbf{m}^\star)P$ and, given that $\tilde{\sigma}^\star$ is a valid designated signature, we have

$$\tilde{\sigma_1}^\star = e(aP' + rK(\mathbf{m}^\star)P, bP) = e(P,P)^{abc} e(K(\mathbf{m}^\star)\sigma_2^\star, bP)$$

and $\sigma_2^\star = rP$ for some $r \in \mathbb{Z}_q^*$, wherefrom $e(P,P)^{abc} = \tilde{\sigma_1}^\star/e(K(\mathbf{m}^\star)\sigma_2^\star, bP)$ is extractable by $\mathcal{B}$.

The simulator $\mathcal{B}$'s probability of success remains to be assessed. We remark that it terminates without aborting if, $J(\mathbf{m}) \neq 0 \bmod q$ for all messages m submitted in a signing query. As $0 \leq k\ell < q$ and $x' + \sum_{i=1}^n m_i x_i < \ell(n+1) < q$, we note that $J(\mathbf{m}) = 0 \bmod q$ implies $J(\mathbf{m}) = 0 \bmod \ell$ (and thus $J(\mathbf{m}) \neq 0 \bmod \ell$ implies $J(\mathbf{m}) \neq 0 \bmod q$). Hence, to simplify the analysis, we may force $\mathcal{B}$ to abort whenever $J(\mathbf{m}) = 0 \bmod \ell$ in a signing query. Besides, $\mathcal{B}$ is successful if the target message happens to satisfy $J(\mathbf{m}^\star) = 0 \bmod q$.

More formally, if $\mathbf{m}_1, \ldots, \mathbf{m}_{q_s}$ are messages appearing in some signing query and if we define the events $A_i : J(\mathbf{m}_i) \neq 0 \bmod \ell$ and $A^\star : J(\mathbf{m}^\star) = 0 \bmod q$, the probability that $\mathcal{B}$ does not fail is $\Pr[\neg\mathsf{abort}] \geq \Pr[\bigwedge_{i=1}^{q_s} A_i \wedge A^*]$. Given that $J(\mathbf{m}^\star) = 0 \bmod q$ implies $J(\mathbf{m}^\star) = 0 \bmod \ell$ and that, if $J(\mathbf{m}^\star) = 0 \bmod \ell$, there is a unique value $k \in \{0, \ldots, n\}$ that yields $J(\mathbf{m}^\star) = 0 \bmod q$, we have $\Pr[A^\star] = \Pr[J(\mathbf{m}^\star) = 0 \bmod \ell]\Pr[J(\mathbf{m}^*) \bmod q | J(\mathbf{m}^\star) = 0 \bmod \ell] = \dfrac{1}{\ell}\dfrac{1}{n+1}$.

Moreover, $\Pr[\bigwedge_{i=1}^{q_s} A_i | A^\star] = 1 - \sum_{i=1}^{q_s} \Pr[\neg A_i | A^\star] = 1 - \frac{q_s}{\ell}$, where the rightmost equality stems from the fact that A_i is independent of $A^\star$ for any i (hence $\Pr[\neg A_i | A^\star] = 1/\ell$). Putting the above together, we find that

$$\Pr[\neg\mathsf{abort}] = \Pr[A^\star]\Pr[\bigwedge_{i=1}^{q_s} A_i | A^*] = \frac{1}{\ell(n+1)}\left(1 - \frac{q_s}{\ell}\right) = \frac{1}{4q_s(n+1)}$$

thanks to the choice of $\ell = 2q_s$. □

4.2 Anonymity

The following theorem states the signer's privacy in a weaker sense than definition 3: verification queries are indeed disallowed throughout the game. The proof follows ideas from [34] and is detailed in the full version of the paper.

Theorem 2. *If an attacker $\mathcal{D}$ is able to $(t, q_s, 0, \varepsilon)$-break the anonymity, there is an algorithm $\mathcal{B}$ that (t', ε')-breaks the DBDH assumption where*

$$\varepsilon' \geq \frac{\varepsilon}{32q_s(n+1)} \qquad t' \leq t + O(q_s\tau_m + \varepsilon^{-2}\ln(\varepsilon^{-1})\mu^{-1}\ln(\mu^{-1}))$$

where τ_m denotes the cost of a scalar multiplication in $\mathbb{G}$.

The next section shows a variant of our scheme where the anonymity property holds in the strong sense of definition 3.

Remark 1. In [25], Lipmaa, Wang and Bao identified a new security requirement for designated verifier signatures: the *non-delegability.* This means that neither the signer nor the designated verifier should be able to produce a "meta-key" which allows to generate new signatures without revealing their secret. Even if this requirement is debatable, our scheme is delegatable (for instance the verifier can publish $\alpha_V P'$). As suggested in [33], delegability is inherent to all UDVS.

5 Avoiding the gap assumption

In this section, we modify our scheme to obtain a variant which is strongly unforgeable under a weaker assumption. This version is obtained using the generic construction of Boneh, Shen and Waters [9] that makes strongly unforgeable any weakly unforgeable signature of some particular kind.

As in [9], we assume that group elements have unique encoding as the scheme would not be strongly unforgeable otherwise.

– $\mathsf{UDVS}.\Sigma.\mathsf{Setup}$ is as in section 3 except that it additionally selects a generator $Q \overset{R}{\leftarrow} \mathbb{G}$. Hash function h is also replaced by a collision-resistant family $[H]_\kappa$ of hash functions $H_\kappa : \{0,1\}^* \to \{0,1\}^n$ indexed by keys $\kappa \in \mathcal{K}$. Public parameters consist of $\mathsf{params} := \{n, q, \mathbb{G}, \mathbb{H}, e, P, P', Q, U', U_1, \dots, U_n, F, [H]_\kappa, \mathcal{K}\}$.

– UDVS.Σ.KeyGen: a signer's private key is a random $\alpha_S \overset{R}{\leftarrow} \mathbb{Z}_q^*$; his public key is made of a group element $P_S = \alpha_S P$ and a key $\kappa \in \mathcal{K}$.
– UDVS.Σ.Sign: given a message $M \in \{0,1\}^*$,
  1. Pick at random $r, s \overset{R}{\leftarrow} \mathbb{Z}_q^*$ and set $\sigma_2 = rP \in \mathbb{G}$.
  2. Compute $t = H_\kappa(M||\sigma_2) \in \{0,1\}^n$ and view it as an element of $\mathbb{Z}_q$.
  3. Compute $\mathbf{m} = H_\kappa(tP + sQ) \in \{0,1\}^n$.
  4. Compute $\sigma_1 = \alpha_S P' + rF(\mathbf{m}) \in \mathbb{G}$.

  The signature is $\sigma = (\sigma_1, \sigma_2, s) = (\alpha_S P' + rF(\mathbf{m}), rP, s)$
– UDVS.Register is as in section 3.
– UDVS.Σ.Verify: given an ordinary signature $\sigma = (\sigma_1, \sigma_2, s)$ on M,
  1. Set $t = H_\kappa(M||\sigma_2) \in \{0,1\}^n$ and view it as an element of $\mathbb{Z}_q$.
  2. Compute $\mathbf{m} = H_\kappa(tP + sQ) \in \{0,1\}^n$ and accept if and only if

$$e(\sigma_1, P) = e(P_S, P')e(\sigma_2, F(\mathbf{m}))$$

– UDVS.VKeyGen is as in section 3.
– UDVS.Designate: to designate a signature $\sigma = (\sigma_1, \sigma_2, s)$ for a verifier V, a signature holder turns it into $\tilde{\sigma} = (\tilde{\sigma}_1, \sigma_2, s)$ with $\tilde{\sigma_1} = e(\sigma_1, P_V)$.
– UDVS.DVerify: given a purported signature $(\tilde{\sigma}_1, \sigma_2, s)$ on M, the verifier computes $t = H_\kappa(M||\sigma_2)$ (which is viewed as an element of $\mathbb{Z}_q$), $\mathbf{m} = H_\kappa(tP + sQ) \in \{0,1\}^n$ and checks whether $\tilde{\sigma_1} = e(P_S, P')^{\alpha_V} e(\sigma_2, F(\mathbf{m}))^{\alpha_V}$

5.1 Security

The present construction has a security proof under the Bilinear Diffie-Hellman assumption which is deemed reasonable by now. However, its strength does not depend on how many signing or verification requests are allowed to adversaries whatsoever. This is a noticeable improvement over [33, 35] and the scheme of section 3. The proof uses a technique which goes back to Ogata *et al.* [22] who showed how to avoid gap assumptions in the security proof [28] of a variant of the Chaum-van Antwerpen undeniable signature [12].

Theorem 3. *If a forger $\mathcal{F}$ can $(t, q_s, q_v, \varepsilon)$-break the strong unforgeability, there exits an algorithm $\mathcal{B}$ that (t', ε')-breaks the BDH assumption where*

$$\varepsilon' \geq \frac{\varepsilon}{12q_s(n+1)(q_v+1)} \qquad t' \leq t + O((q_s + q_v)\tau_m + q_v\tau_p)$$

and τ_m, τ_p stand for the same quantity as in theorem 1.

The key idea is that, unless the scheme is not strongly existentially unforgeable, all verification queries necessarily involve signatures that were obtained from signing oracles or that are invalid. The simulator's strategy is to guess which verification query involves a forged signature and reject signatures involved in all other queries. Such a proof strategy does not apply to our first UDVS scheme where signatures obtained from a signing oracle may be publicly turned into other signatures on the same messages.

Proof. Algorithm $\mathcal{B}$ combines the technique of [9] with a strategy introduced in [22] to prove the security of a variant of the Chaum-van Antwerpen [12] undeniable signature under the CDH assumption. In the simulation, $\mathcal{B}$ maintains a history L_S of all signing queries and their outputs. Whenever $\mathcal{F}$ asks for a plain signature, $\mathcal{B}$ also computes and stores in L_S the unique (recall that designation is deterministic) matching designated signature for the target verifier $P_{V^\star}$.

As in theorem 1 of [9], the forger makes her signing queries on messages $M_1, \ldots, M_{q_s}$ that result in a list of triples $(\tilde{\sigma_{i,1}}, \sigma_{i,2}, s_i)$ for $i = 1, \ldots, n$. Let $t_i = H_\kappa(M_i||\sigma_{2,i})$ and $w_i = t_iP + s_iQ$. Let also $\langle M^\star, (\tilde{\sigma_1}^\star, \sigma_2^\star, s^\star)\rangle$ be the fake designated signature produced by $\mathcal{F}$ and $t^\star = H_\kappa(M^\star||\sigma_2^\star)$, $w^\star = t^\star P + s^\star Q$. Just like the proof of theorem 1 in [9], we distinguish three kinds of forgeries:

Type I: a forgery with $w^\star = w_i$ and $t^\star = t_i$ for some $i \in \{1, \ldots, q_s\}$.
Type II: a forgery with $w^\star = w_i$ and $t^\star \neq t_i$ for some $i \in \{1, \ldots, q_s\}$.
Type III: a forgery for a new element $w^\star \neq w_i$ for any $i \in \{1, \ldots, q_s\}$.

A successful forger comes with a forgery of Type I, Type II or Type III and $\mathcal{B}$ has to guess which kind of forger $\mathcal{F}$ will be at the outset of the simulation.

In all cases, $\mathcal{F}$ is allowed making up to q_v verification queries on triples $\tilde{\sigma}_j = (\tilde{\sigma_{j,1}}, \sigma_{j,2}, s)$ which are likely to be designated signatures intended to the target verifier $V^\star$ and bearing the name of the target signer $S^\star$. The main difficulty for $\mathcal{B}$ is to deal with those queries without resorting to a decision oracle. For convenience, $\mathcal{F}$'s forgery is viewed as her $q_v + 1^{th}$ query to the verification oracle. A verification request $(M_j, \tilde{\sigma}_j)$, with $j \in \{1, \ldots, q_v + 1\}$, is called *special* if $\tilde{\sigma}_j$ is a valid signature on M_j for signer $S^\star$ and designated verifier $V^\star$ and if it does not appear in $\mathcal{B}$'s history L_S of signing queries. Clearly, a *special* verification query is a breach (which is assumed to occur at least once in a real attack) in the strong unforgeability property. Before the simulation starts, $\mathcal{B}$ has to guess the index $j^\star \in \{1, \ldots, q_v + 1\}$ of the first special query.

Upon reception of a verification query $(M_j, \tilde{\sigma}_j)$, $\mathcal{B}$ distinguishes two cases

- if $j < j^\star$, $\mathcal{B}$ declares the signature as 'invalid' if $(M_j, \tilde{\sigma}_j)$ does not appear in the history L_S. Otherwise, it returns 'valid'.
- if $j = j^\star$, $\mathcal{B}$ aborts if $(M_{j^\star}, \tilde{\sigma}_{j^\star})$ appears in L_S (which means that $\mathcal{B}$ failed to guess the index of the first special query). Otherwise, $\mathcal{B}$ halts and bets that $(M_{j^\star}, \tilde{\sigma}_{j^\star})$ is indeed an existential forgery of either Type I, Type II or Type III . In this desired event, the BDH solution is extracted as explained below.

If signing queries are correctly answered, a sufficient condition for $\mathcal{B}$ to perfectly simulate the verification oracle is to correctly guess the index $j^\star$ of the first special verification request. This obviously happens with probability $1/(q_v + 1)$.

We now explain how $\mathcal{B}$ solves a BDH instance (aP, bP, cP) using $\mathcal{F}$. It first chooses $c_{mode} \in \{1, 2, 3\}$ in an attempt to foresee which kind of forger $\mathcal{F}$ will be.

- If $c_{mode} = 1$, $\mathcal{B}$ bets on a Type I forgery which is easily seen to break the collision-resistance of $[H]_\kappa$. A random key $k \in \mathcal{K}$ is chosen by $\mathcal{B}$ that generates the remaining public key components following the specification of the protocol. All queries are dealt with using the relevant private elements. When $\mathcal{F}$ outputs a forgery $\langle M^\star, \tilde{\sigma}^\star = (\tilde{\sigma}_1^\star, \sigma_2^\star, s^\star)\rangle$, we have $t^\star = H_\kappa(M^\star||\sigma_2^\star) = H_\kappa(M_i||\sigma_{i,2}) = t_i$ and $w^\star = t^\star P + s^\star Q = t_i P + s_i Q = w_i$ for some $i \in \{1, \ldots, q_s\}$. Hence, we must also have $s^\star = s_i$. Assuming that $M^\star||\sigma_2^\star = M_i||\sigma_{i,2}$, we should have $\tilde{\sigma}_1^\star \neq \sigma_{i,1}^\star$ (as $\tilde{\sigma}^\star$ would not be a forgery otherwise) which is impossible as $\tilde{\sigma_1}^\star$ is uniquely determined by $t^\star$, $\sigma_2^\star$ and $s^\star$ if $\tilde{\sigma}^\star$ is valid. Therefore, we have a collision $H_\kappa(M^\star||\sigma_2^\star) = H_\kappa(M_i||\sigma_{i,2})$ with $M^\star||\sigma_2^\star \neq M_i||\sigma_{i,2}$.

- If $c_{mode} = 2$, $\mathcal{B}$ expects a Type II forgery and prepares public parameters with $Q = aP$ being part of the input of its BDH instance. The other public parameters and public key components are generated following the protocol. All adversarial queries are answered using the relevant private keys. As $\mathcal{F}$ comes with her forgery $\langle M^\star, \tilde{\sigma}^\star = (\tilde{\sigma}_1^\star, \sigma_2^\star, s^\star)\rangle$, we have $w^\star = t^\star P + s^\star Q = t_i P + s_i Q = w_i$ with $t^\star = H_\kappa(M^\star||\sigma_2^\star) \neq H_\kappa(M_i||\sigma_{i,2}) = t_i$. This allows $\mathcal{B}$ to extract $a = (t_i - t^\star)/(s^\star - s_i)$ and thereby solve the BDH problem by computing $e(bP, cP)^a$.
- If $c_{mode} = 3$, $\mathcal{B}$ expects a forgery on a new "message" $w^\star$ and proceeds in the same way as the simulator of theorem 1.

When assessing $\mathcal{B}$'s advantage, we already observed that it correctly guesses the index of the first special verification query with probability $1/(q_v + 1)$. As it succeeds in foresee the right kind of forgery with probability $1/3$, the lower bound on its advantage easily follows from theorem 1. $\square$

Strong unforgeability also implies a provable anonymity in the strict sense of definition 3. The proof of the following theorem is very similar to the one of theorem 2. By virtue of strong unforgeability, all verification queries pertain to designated signatures that are either invalid or that result from a signing query. Hence, for each verification query, the simulator just has to compare the candidate signature to those it returned when dealing with signing queries.

Theorem 4. *If an attacker $\mathcal{D}$ can $(t, q_s, q_v, \varepsilon)$-break the anonymity, there is an algorithm $\mathcal{B}$ that (t', ε')-breaks the DBDH assumption where*

$$\varepsilon' \geq \frac{\varepsilon}{32 q_s (n+1)} \qquad t' \leq t + O(q_s \tau_m + q_v \tau_p + \varepsilon^{-2} \ln(\varepsilon^{-1}) \mu^{-1} \ln(\mu^{-1}))$$

where τ_m, τ_p denotes the same quantity as in theorem 1.

6 Conclusion

We proposed the first UDVS schemes which are secure under reasonable complexity assumptions in the standard model where our constructions are also the only ones to achieve anonymity in the sense of [23].

The next table compares our two schemes with other secure systems in the standard model. Our constructions appear to be competitive with [33,35]. Their main drawback remains the size of public parameters. We leave open the problem of finding UDVS schemes that are secure under mild assumptions in the standard model without using large public parameters. A trick independently suggested in [11,26] allows for a step towards this purpose.

Schemes	[ZFI05]	[Ver06]	Scheme 1	Scheme 2
Assumptions	p-SDH + KEA	p-SDH + KEA	wGBDH	BDH
Sign	1 $\exp_{\mathbb{G}_1}$	1 $\exp_{\mathbb{G}_1}$	2 $\exp_{\mathbb{G}_1}$	4 $\exp_{\mathbb{G}_1}$
Verify	1 P + 1 m-$\exp_{\mathbb{G}_2}$	1 P + m-$\exp_{\mathbb{G}_2}$	1 PP† + 1 $\exp_{\mathbb{G}_2}$	1 PP† + 1 $\exp_{\mathbb{G}_2}$
Designate	1 P + 2 $\exp_{\mathbb{G}_2}$	3 $\exp_{\mathbb{G}_1}$	1 P	1 P
DVerify	2 P + 2 $\exp_{\mathbb{G}_2}$	2 PP + 1 m-$\exp_{\mathbb{G}_2}$	1 P + 1 $\exp_{\mathbb{G}_2}$	1 P + 1 $\exp_{\mathbb{G}_2}$
Size	320	320	320	480
DSize	2400*	640	480*	640*

P: pairing operation, PP: product of 2 pairings, m-exp: multi-exponentiation with 2 exponents

($\dagger$) In both of our schemes, we assume that $e(P_S, P')$ is stored as part of the signer's public key.

($\star$) These sizes can be obtained using asymmetric pairings and curves of embedding degree 12 [3] with compression [2].

References

1. J.-H. An, Y. Dodis, T. Rabin. On the security of joint signature and encryption. Proc. of Eurocrypt'02, Springer LNCS vol. 2332, 83–107 (2002).
2. P. S. L. M. Barreto, M. Scott. Compressed Pairings. Proc. of Crypto'04, Springer LNCS vol. 3152, 140–156 (2004).
3. P. S. L. M. Barreto, M. Naehrig. Pairing-Friendly Elliptic Curves of Prime Order. Proc. of SAC'05, Springer LNCS vol. 3897, 319–331 (2005).
4. M. Bellare, A. Palacio. The Knowledge-of-Exponent Assumptions and 3-Round Zero-Knowledge Protocols. Proc. of Crypto'04, Springer LNCS vol. 3152, 273-289 (2004)
5. M. Bellare, P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. Proc. of ACM CCS'93, ACM Press, 62–73 (1993)
6. A. Bender, J. Katz, R. Morselli. Ring Signatures: Stronger Definitions, and Constructions without Random Oracles. Proc. of TCC'06. Springer LNCS vol. 3876, 60–79 (2006)
7. D. Boneh, X. Boyen. Short Signatures Without Random Oracles. Proc. of Eurocrypt'04, Springer LNCS vol. 3027, 56–73 (2004)
8. D. Boneh, M. Franklin. Identity-based encryption from the Weil pairing. Proc. of Crypto'01, Springer LNCS vol. 2139, 213–229 (2001)
9. D. Boneh, E. Shen, B. Waters, Strongly Unforgeable Signatures Based on Computational Diffie-Hellman. Proc. of PKC'05, Springer LNCS vol. 3958, 229–240 (2005).
10. R. Canetti, O. Goldreich, S. Halevi. The random oracle methodology, revisited. Journal of the ACM 51(4), 557–594 (2004)
11. S. Chatterjee, P. Sarkar. Trading time for space: Towards an efficient IBE scheme with short(er) public parameters in the standard model. Proc. of ICISC 2005, Springer LNCS vol. 3935, 424–440 (2006)
12. D. Chaum, H. van Antwerpen. Undeniable Signatures, Proc. of Crypto'89, Springer LNCS vol. 435, 212–216 (1989).
13. J. H. Cheon. Security Analysis of the Strong Diffie-Hellman Problem. Proc. of Eurocrypt'06, Springer LNCS vol. 4004, 1–11 (2006)
14. I. Damgård. Towards practical public-key cryptosystems provably-secure against chosen-ciphertext attacks. Proc. of Crypto'91, Springer LNCS vol. 576, 445–456 (1991)

15. A. Dent. The Hardness of the DHK Problem in the Generic Group Model. Cryptology ePrint Archive: report 2006/156(2006)
16. A. Fiat, A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. Proc. of Crypto'86. Springer LNCS vol. 263, 186–194 (1986)
17. S. Galbraith, W. Mao: Invisibility and Anonymity of Undeniable and Confirmer Signatures. Proc. of CT-RSA 2003, Springer LNCS vol. 2612, 80–97 (2003)
18. S. Goldwasser, S. Micali, R. L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. SIAM J. Comput. 17(2), 281–308 (1988)
19. S. Hada, T. Tanaka. On the Existence of 3-Round Zero-Knowledge Protocols. Proc. of Crypto'98. Springer LNCS vol. 1462, 408–42 (1998)
20. M. Jakobsson, K. Sako, R. Impagliazzo: Designated Verifier Proofs and their Applications. Proc. of Eurocrypt'96, Springer LNCS vol. 1070, 142–154 (1996)
21. A. Joux. A one round protocol for tripartite Diffie-Hellman. Proc. of ANTS-IV, Springer LNCS vol. 1838, 385–394 (2000)
22. W. Ogata, K. Kurosawa, S.-H. Heng. The Security of the FDH Variant of Chaum's Undeniable Signature Scheme. Proc of PKC'05, Springer LNCS vol. 3386, 328–345 (2005)
23. F. Laguillaumie, D. Vergnaud. Designated Verifiers Signature: Anonymity and Efficient Construction from *any* Bilinear Map. Proc. of SCN'04, Springer LNCS vol. 3352, 107–121 (2005)
24. F. Laguillaumie, D. Vergnaud. Multi-Designated Verifiers Signature Schemes. Proc. of ICICS'04, Springer LNCS vol. 3269, 495–507 (2004)
25. H. Lipmaa, G. Wang, F. Bao. Designated Verifier Signature Schemes: Attacks, New Security Notions and A New Construction. Proc. of ICALP 2005, Springer LNCS vol. 3580, 459–471 (2005)
26. D. Naccache. Secure and *Practical* Identity-Based Encryption. Cryptology ePrint Archive : report 2005/369 (2005)
27. C. Y. Ng, W. Susilo, Y. Mu. Universal Designated Multi Verifier Signature Schemes. Proc. of SNDS 2005, IEEE Press, 305–309 (2005)
28. T. Okamoto, D. Pointcheval: The Gap-Problems: a New Class of Problems for the Security of Cryptographic Schemes. Proc. of PKC'01 Springer LNCS vol. 1992, 104–118 (2001)
29. S. Saeednia, S. Kremer, O. Markowitch. An Efficient Strong Designated Verifier Signature Scheme. Proc. of ICISC 2003, Springer LNCS vol. 2836, 40–54 (2003)
30. C. P. Schnorr. Efficient identification and signatures for smart cards. Proc. of Crypto'89, Springer LNCS vol. 435, 239–252 (1989)
31. R. Steinfeld, L. Bull, H. Wang, J. Pieprzyk. Universal Designated Verifier Signatures. Proc. of Asiacrypt'03, Springer LNCS vol. 2894, 523–542 (2003)
32. R. Steinfeld, H. Wang, J. Pieprzyk. Efficient Extension of Standard Schnorr/RSA signatures into Universal Designated-Verifier Signatures. Proc. of PKC'04, Springer LNCS vol. 2947, 86–100 (2004)
33. D. Vergnaud. New extensions of Pairing-based Signatures into Universal Designated Verifier Signatures. To appear in Proc. of ICALP 2006.
34. B. Waters. Efficient Identity-Based Encryption Without Random Oracles. Proc. of Eurocrypt'05. Springer LNCS vol. 3494, 114–127 (2005)
35. R. Zhang, J. Furukawa, H. Imai. Short signature and Universal Designated Verifier Signature without Random Oracles. Proc. of ACNS'05, Springer LNCS vol. 3531, 483–498 (2005)

A Waters's Signature Scheme

This section recalls the description of Waters's signature [34] which is existentially unforgeable under the Diffie-Hellman assumption in pairing-friendly groups.

- KeyGen: this algorithm chooses groups $(\mathbb{G},\mathbb{H})$ equipped with a bilinear map $e:\mathbb{G}\times\mathbb{G}\rightarrow\mathbb{H}$, and a collision-resistant hash function $h:\{0,1\}^*\rightarrow\{0,1\}^n$. A secret integer $\alpha \overset{R}{\leftarrow} \mathbb{Z}_q^*$ is picked at random, as well as three random elements P, P' and U of $\mathbb{G}$ and a random n-tuple $\overline{U}=(U_1,\dots,U_n)\in\mathbb{G}^n$. The public key is then $(P,P',P_S,U',\overline{U})$ with $P_S=\alpha P$. The secret key is α.
- Sign: let $\mathbf{m}=h(M)$, for a message $M\in\{0,1\}^*$, $\mathbf{m}=m_1\dots m_n$, with $m_i\in\{0,1\}$ for all $i\in\{1,\dots,n\}$, and $\mathcal{M}$ be the set of indexes $i\in\{1,\dots,n\}$ s.t. $m_i=1$. A signature of M is produced by choosing $r\overset{R}{\leftarrow}\mathbb{Z}_q^*$ and setting $\sigma=(\sigma_1,\sigma_2)$ with $\sigma_1=\alpha P'+r(U'+\sum_{i\in\mathcal{M}}U_i)$ and $\sigma_2=rP$.
- Verify: a purported signature σ on M is accepted if on only if $e(\sigma_1,P)=e(P_S,P')e(\sigma_2,U'+\sum_{i\in\mathcal{M}}U_i)$.

The disadvantage of the scheme is the length of the public key which contains more than 160 group elements for typical parameters. However, the vector $(U',U_1,\dots,U_n)$ can come from a common reference string and be shared by several signers in a system.

B Proof of Theorem 2

Algorithm $\mathcal{B}$ is fed with the description of a group $\mathbb{G}$ together with a generator P and a tuple $(aP,bP,cP,h)\in\mathbb{G}^3\times\mathbb{H}$. It has to decide if $h=e(P,P)^{abc}$. To do so, it performs a simulation which is quite similar to the one in the security proof of Waters's scheme [34].

Setup: The preparation phase is proceeds almost exactly as in theorem 1. More precisely, $\mathcal{B}$ sets $P'=cP$ but it defines verifier and signers' public keys as $P_V=bP$, $P_{S,0}=(aP)+\rho_0P$ and $P_{S,1}=(aP)+\rho_1P$ respectively for random values $\rho_0,\rho_1\overset{R}{\leftarrow}\mathbb{Z}_q^*$. Another difference lies in the optimal value of ℓ which is set to $\ell=4q_s$. All group elements $U',U_1,\dots,U_n$ are chosen as in the proof of theorem 1.

Signing queries: as in the proof of theorem 1, the simulator aborts if a signing query is made on a message $\mathbf{m}=h(M)$ for which $J(\mathbf{m})=0 \bmod q$. Otherwise, a valid signature can be computed on all messages $\mathbf{m}$ for which $J(\mathbf{m})\neq 0 \bmod q$. If the signature is requested for signer $P_{S,b}$, with $b\in\{0,1\}$, $\mathcal{B}$ picks $r\overset{R}{\leftarrow}Z_q$ and computes

$$\sigma=(\sigma_1,\sigma_2)=\left(-\frac{K(\mathbf{m})}{J(\mathbf{m})}(aP)+rF(\mathbf{m})+\rho_bP',-\frac{1}{J(\mathbf{m})}(aP)+rP\right).$$

If we define $\tilde{r} = r - a/J(\mathbf{m})$, σ is a correct as

$$\begin{aligned}\sigma_1 &= -\frac{K(\mathbf{m})}{J(\mathbf{m})}(aP) + rF(\mathbf{m}) + \rho_b P' \\ &= -\frac{K(\mathbf{m})}{J(\mathbf{m})}(aP) + \tilde{r}F(\mathbf{m}) + \frac{a}{J(\mathbf{m})}(J(\mathbf{m})P' + K(\mathbf{m})P) + \rho_b P' \\ &= (a + \rho_b)P' + \tilde{r}F(\mathbf{m})\end{aligned}$$

and $\sigma_2 = (r - a/J(\mathbf{m}))P = \tilde{r}P$. The resulting signature is then passed to the public designation procedure.

Challenge: after a number of queries, $\mathcal{F}$ enters the challenge phase and outputs a message $\mathbf{m}^\star = h(M^\star)$ on which he wants to be challenged. At that point, $\mathcal{B}$ halts and reports "failure" if $J(\mathbf{m}^\star) \neq 0 \bmod q$. Otherwise, $F(\mathbf{m}^\star) = K(\mathbf{m}^\star)P$ and $\mathcal{B}$ flips a fair coin $b^\star \in \{0,1\}$. It returns the challenge $\tilde{\sigma}^\star = (\tilde{\sigma}_1^\star, rP)$ where

$$\tilde{\sigma_1}^\star = h \cdot e(cP, bP)^{\rho_{b^\star}} \cdot e(K(\mathbf{m}^\star)rP, bP)$$

and $\sigma_2^\star = rP$ for a randomly chosen $r \overset{R}{\leftarrow} \mathbb{Z}_q^*$. Observe that, if $h = e(P,P)^{abc}$, we have

$$\begin{aligned}\tilde{\sigma_1}^\star &= e(acP, bP) \cdot e(\rho_{b^\star}P', bP) \cdot e\left(K(\mathbf{m}^\star)rP, bP\right) \\ &= e\left((a + \rho_{b^\star})P' + rK(\mathbf{m}^\star)P, bP\right) \\ &= e\left((a + \rho_{b^\star})P' + rF(\mathbf{m}^\star), bP\right)\end{aligned}$$

and $\tilde{\sigma}^\star$ appears as a valid designated signature bearing the name of signer $P_{S,b^\star}$. In contrast, if h is random in $\mathbb{H}$, $\tilde{\sigma}^\star$ is independent of the bit $b^\star$.

Guess: after another series of queries, $\mathcal{D}$ outputs a bit $b \in \{0,1\}$.

Artifical abort: at this point, the simulator cannot directly use the adversary's output as the latter's probability of success could be correlated with the probability that the simulator needs to abort. This is due to the fact that two distinct sets of q_s signing queries may cause an abort with different probabilities. The simulator corrects this by imposing all possible sets of queries to cause an abort of the simulation with (almost) the same probability $1 - \mu$ which is a lower bound on any set of signing queries causing an abort before the guess stage.

Let $\overrightarrow{\mathbf{M}} = \mathbf{m}_1, \ldots, \mathbf{m}_{q_s}$ denote messages that are input of signing queries in phases 1 and 2. Let $\mathbf{m}^\star = h(M^\star)$ be the challenge message (all these values are defined at this step of the simulation). If X' is a set of simulation values $x', x_1, \ldots, x_n \in \mathbb{Z}_\ell$, we define the function $\varphi(X', \overrightarrow{\mathbf{M}}, \mathbf{m}^\star)$ as

$$\begin{aligned}\varphi(X', \overrightarrow{\mathbf{M}}, \mathbf{m}^\star) = 0 \text{ if } (\bigwedge_{i=1}^{q_s} J(\mathbf{m}_i) \neq 0 \bmod \ell) \wedge J(\mathbf{m}^\star) = 0 \bmod q \\ 1 \text{ otherwise}\end{aligned}$$

which evaluates to 0 if signing and challenge queries do not cause the simulation to abort for a given choice X' of simulation values. For a given set of queries $(\overrightarrow{\mathbf{M}}, \mathbf{m}^\star)$, we consider the following probability over the simulation values $\eta = \Pr[\varphi(X', \overrightarrow{\mathbf{M}}, \mathbf{m}^\star) = 0]$.
In order to compute an estimation η' of η, the simulator has to sample $O(\varepsilon^{-2} \ln(\varepsilon^{-1})\mu^{-1} \ln(\mu^{-1}))$ times the probability η by choosing a random X' and evaluating $\varphi(X', \overrightarrow{\mathbf{M}}, \mathbf{m}^\star)$ (the sampling does not require to run the adversary again). Let $\mu = \frac{1}{8nq_S}$ denote a lower bound on the probability of not aborting for any set of queries (we refer to Waters's proof [34] for details on how to calculate μ). If $\eta' \geq \mu$, the simulator artificially aborts with probability $\frac{\eta' - \mu}{\eta'}$ (and not abort with probability $\frac{\lambda}{\eta'}$) and produces a random guess $\beta' \xleftarrow{R} \{0, 1\}$.

Output: If $\mathcal{B}$ did not abort, it returns $\beta' = 1$ (meaning that its input is a valid BDH tuple) if $b = b^\star$. Otherwise, it bets being in front of a random tuple and returns $\beta' = 0$.

This completes the description of the simulation. We refer to [34] for a complete analysis of the bounds of the reduction. □