Multi-Designated Verifiers Signatures

Fabien Laguillaumie[1,2] and Damien Vergnaud[2]

[1] France Télécom R&D
42, rue des Coutures, B.P. 6243, 14066 Caen Cedex 4, France,
[2] Laboratoire de Mathématiques Nicolas Oresme
Université de Caen, Campus II, B.P. 5186,
14032 Caen Cedex, France,
{laguillaumie,vergnaud}@math.unicaen.fr

Abstract. Designated verifier signatures were introduced in the middle of the 90's by Jakobsson, Sako and Impagliazzo, and independenty patended by Chaum as private signatures. In this setting, a signature can only be verified by a unique and specific user. At Crypto'03, Desmedt suggested the problem of generalizing the designated verifier signatures. In this case, a signature should be intended to a specific set of different verifiers. In this article, we provide a formal definition of multi-designated verifiers signatures and give a rigorous treatment of the security model for such a scheme. We propose a construction based on ring signatures, which meets our definition, but does not achieve the *privacy of signer's identity* property. Finally, we propose a very efficient bi-designated verifiers signature scheme based on bilinear maps, which protects the anonymity of signers.

Keywords: multi-designated verifiers signatures, ring signatures, bilinear maps, privacy of signer's identity, exact security.

1 Introduction.

At Crypto'03 rump session [8], Desmedt raised the problem of generalizing the *designated verifier signatures* (DVS) concept, introduced independently by Chaum in 1996 in the patent [7] and by Jakobsson, Sako and Impagliazzo in [11]. In this model, the signature of a message is intended to a specific verifier, chosen by the signer, who will be the only one able to verify its validity. As pointed out in [15], this can be viewed as a "light signature scheme". No one else than the designated person can be convinced by this signature because he can also perform the signature by himself. In particular, it does not provide the main property of usual signature scheme: the non-repudiation. Such signature schemes have numerous applications in call for tenders, electronic voting or electronic auction. The question opened by Desmedt was to allow several designated verifiers. This new primitive that we call *multi-designated verifiers signatures* (MDVS) may have many interests in a multi-users setting, for instance it seems promising for the design of fair distributed contract signing.

As early as their paper [11], Jakobsson, Sako and Impagliazzo suggested an extension of their protocol to multiple designated verifiers. As their single designated verifier scheme, it did not catch the notion of *privacy of signer's identity*, introduced in [13], without an additional encryption layer.

Our contributions We propose a construction of multi-designated verifiers signatures where the signer chooses to sign a message for a fixed numbers of specific designated verifiers. Basically, the main security properties that we want to achieve are, in addition to the unforgeability, the *source hiding* and the *privacy of signer's identity*. These notions are formally defined in the section 2.2. Our construction is based on the notion of ring signatures, defined in [15] by Rivest, Shamir and Tauman. The idea of such a protocol is to produce a signature which has the property that any verifier is convinced that this signature has been done by one member of a set of users, but is not able to determine which one. Our scheme can be instanciated with the ring signature introduced in [5] by Boneh, Gentry, Shacham and Lynn, or in [18] by Zhang, Safavi-Naini and Susilo, based on bilinear maps as well as those proposed in [10] by Herranz and Sáez, or in [1] by Abe, Ohkubo and Suzuki based on Schnorr signatures. Contrary to Desmedt's suggestion in his talk [8], the size of our signatures does not grow with the number of verifiers. Unfortunately, in all cases, there is an encryption layer to achieve the notion of privacy of signer's identity. Therefore, there is a need for an n-party key agreement protocol for this encryption scheme. In this case, the protocol looses its spontaneity and becomes less efficient. Finally, we propose an efficient bi-designated verifiers protocol based on bilinear maps, which takes advantage of Joux's tripartite secret exchange [12], thanks to these pairings. In this particular case, there is no need for the supplementary encryption layer to protect the anonymity of signers. The tripartite setting is of recurrent interest in cryptography, and this scheme may find many applications. We propose a formal definition for the security of such protocols. We prove that our schemes are secure against existential forgery and do not reveal the signer's identity under a chosen message attack in the random oracle model.

2 Multi-Designated Verifiers Signatures.

In this section, we define the concept of multi-designated verifiers signatures and propose a formal model of security for such a scheme.

2.1 Definition

Definition 1 (Weak Multi-Designated Verifier Signature Scheme). *Let k and n be two integers, a* weak n-designated verifiers signature scheme `MDVS` *with security parameter k is defined by the following:*

- *a* **setup algorithm** `MDVS.Setup`*: it is a probabilistic algorithm which takes as input a security parameter k and outputs the public parameters,*

– *a* **key generation algorithm for signers** *MDVS.SKeyGen: it is a probabilistic algorithm which takes as input the public parameters and an entity*[1] *A, and outputs a pair of keys (pk_A, sk_A),*
– *a* **key generation algorithm for the designated verifiers** *MDVS.VKeyGen: it is a probabilistic algorithm which takes as input the public parameters, an entity B, and outputs a pair of keys (pk_B, sk_B),*
– *an* **n-designated verifiers signing algorithm** *MDVS.Sign: it is an algorithm which takes as input a message m, a signing secret key sk_A, the n verifying public keys of the n entities B_i, $i \in [\![1, n]\!]$ and the public parameters, and outputs a $(B_1, \ldots, B_n)$-designated verifier signature σ of m. This algorithm can be either probabilistic or deterministic,*
– *an* **n-designated verifying algorithm** *MDVS.Verify: it is a deterministic algorithm which takes as input a bit string σ, a message m, a signing public key pk_A, a verifying secret key sk_{B_i}, for some $i \in [\![1, n]\!]$, and the public parameters and tests whether σ is a valid $(B_1, \ldots, B_n)$-designated verifiers signature of m with respect to the keys $pk_A, pk_{B_1}, \ldots, pk_{B_n}$.*

It must satisfy the following properties:

1. **correctness:** *a properly formed $(B_1, \ldots, B_n)$-designated verifiers signature must be accepted by the verifying algorithm. Moreover, a putative signature is accepted by the verifying algorithm using one verifying secret key if and only if it is accepted using each verifying secret key;*
2. **unforgeability:** *given an entity A, it is computationally infeasible, without the knowledge of the secret key of either A or those of all the B_i, $i \in [\![1, n]\!]$, to produce a $(B_1, \ldots, B_n)$-designated verifiers signature that is accepted by the verifying algorithm;*
3. **source hiding:** *given a message m and a $(B_1, \ldots, B_n)$-designated verifiers signature σ of this message, it is (unconditionally) infeasible to determine who from the original signer or the designated verifiers all together performed this signature, even if all secrets are known.*

In [11], Jakobsson *et al.* suggested a stronger notion of anonymity:

Definition 2 (Strong Multi-Designated Verifier Signature Scheme). *Given two integers n and k, a* strong n-designated verifier signature scheme *MDVS with security parameter k, is an n-designated verifier signature scheme with security parameter k, which satisfies the following additional property:*

4. **privacy of signer's identity:** *given a message m and a $(B_1, \ldots, B_n)$-designated verifier signature σ of m, it is computationally infeasible, without the knowledge of the secret key of one B_i for some $i \in [\![1, n]\!]$ or those of the signer, to determine which pair of signing keys was used to generate σ.*

[1] Formally speaking, entities are modelled by probabilistic interactive Turing machines

2.2 Security Model

In this article, the proofs of security are carried in the random oracle model, proposed by Bellare and Rogaway in [2]. Let $B = \{B_i, i = 1, \dots, n\}$ be a group of n entities (the designated verifiers), k be an integer and $\mathtt{MDVS}$ be a n-designated verifiers signature scheme with security parameter k.

Security against existential forgery under chosen message attack. For digital signatures, the strongest security notion was defined by Goldwasser, Micali and Rivest in [9] as *existential forgery against adaptive chosen message attack* (EF-CMA). In the MDVS setting, an EF-CMA-adversary $\mathcal{A}$ is given the n public keys of the B_i's, as well as an access to the random oracle(s) $\mathcal{H}$ and to a signing oracle Σ. As $\mathcal{A}$ cannot verify a signature by himself, one may give him an access to a verifying oracle to check the validity of signatures, as for single designated verifier signatures [17]. On the other hand, during the attack we allow the attacker to corrupt up to $n-1$ designated verifiers (and to do so adaptively), *i.e.* he has access to a corrupting oracle Ξ to obtain the secret information of the corresponding corrupted verifier. Therefore he is able to verify by himself a signature, and we can omit the verifying oracle. $\mathcal{A}$ is allowed to query the signing oracle on the challenge message m but is supposed to output a signature of the message m not given by Σ.

Definition 3 (Security against existential forgery). *Let B be n entities, k and t be integers and ε be a real in $[0,1]$, let MDVS be an n-designated verifiers signature scheme with security parameter k. Let $\mathcal{A}$ be an EF-CMA-adversary against MDVS. We consider the following random experiment:*

$$\boxed{\textit{Experiment } \mathbf{Exp}^{\mathsf{ef-cma}}_{\mathit{MDVS},\mathcal{A}}(k)}$$

$$\begin{array}{l}
\text{params} \xleftarrow{R} \mathit{MDVS.Setup}(k) \\
\textit{For } i = 1, \dots, n \textit{ do } (pk_{B_i}, sk_{B_i}) \xleftarrow{R} \mathit{MDVS.VKeyGen}(\text{params}, B_i) \\
(pk_A, sk_A) \xleftarrow{R} \mathit{MDVS.SKeyGen}(\text{params}, A) \\
(m, \sigma) \leftarrow \mathcal{A}^{\mathcal{H},\Sigma,\Xi}(\text{params}, pk_{B_1}, \dots, pk_{B_n}, pk_A) \\
\textit{Return } \bigvee_{i=1}^{n} \mathtt{MDVS}.\mathit{Verify}(\text{params}, m, \sigma, pk_A, sk_{B_i})
\end{array}$$

We define the success of the adversary $\mathcal{A}$, via $\mathbf{Succ}^{\mathit{ef\text{-}cma}}_{\mathit{MDVS},\mathcal{A}}(k) = Pr\left[\mathbf{Exp}^{\mathit{ef\text{-}cma}}_{\mathit{MDVS},\mathcal{A}}(k) = 1\right]$. *MDVS is said to be (k, t, ε)-EF-CMA secure, if no adversary $\mathcal{A}$ running in time t has a success $\mathbf{Succ}^{\mathit{ef\text{-}cma}}_{\mathit{MDVS},\mathcal{A}}(k) \geq \varepsilon$.*

Source hiding. As argued by the authors in [13], it is desirable, for DVS, to unconditionally protect the identity of the signer, as in a ring signature setting. We refer the reader to [15] for considerations about this property.

Privacy of signer's identity under chosen message attack. We modify the notion of *privacy of signer's identity* introduced in [13] for DVS to fit in the multi-designated verifiers signatures setting. As in the forgery model, the security is also against a chosen message attack (PSI-CMA). If an adversary is given two keys including the one which generates the pair (m, σ), then the possession of this pair (m, σ) should not give him an advantage in determining under which of the two keys the signature was created. We consider a PSI-CMA-adversary $\mathcal{A}$ that runs in two stages. In the `find` stage, it takes two public keys pk_0 and pk_1 and outputs a message $m^\star$ together with some state information $\mathcal{I}^\star$. In the `guess` stage it gets a challenge signature $\sigma^\star$ formed by signing at random the message $m^\star$ under one of the two keys, and must say which key was chosen. In the case of CMA, the adversary has access to the signing oracles Σ_0, Σ_1, to the verifying oracle Υ, and to the random oracle $\mathcal{H}$. The only restriction of the attacker is that he cannot query the pair $(m^\star, \sigma^\star)$ on the verifying oracle.

Definition 4 (Privacy of signer's identity). *Let B be a set of n entities, k and t be integers and ε be a real in $[0,1]$. Let MDVS be an n-designated verifiers signature scheme with security parameter k, and let $\mathcal{A}$ be a PSI-CMA-adversary against MDVS. We consider the following random experiment, for $r \in \{0,1\}$:*

$\boxed{\textit{Experiment } \mathbf{Exp}^{\textsf{psi-cma}-r}_{\textit{MDVS},\mathcal{A}}(k)}$

params $\xleftarrow{R}$ $\texttt{MDVS.Setup}(k)$
For $i = 1, \ldots, n$ *do* $(pk_{B_i}, sk_{B_i}) \xleftarrow{R} \texttt{MDVS.VKeyGen}(\text{params}, B_i)$
$(pk_{A_0}, sk_{A_0}) \xleftarrow{R} \texttt{MDVS.SKeyGen}(\text{params}, A_0)$
$(pk_{A_1}, sk_{A_1}) \xleftarrow{R} \texttt{MDVS.SKeyGen}(\text{params}, A_1)$
$(m^\star, \mathcal{I}^\star) \leftarrow \mathcal{A}^{\mathcal{H}, \Sigma_0, \Sigma_1, \Upsilon}(\texttt{find}, \text{params}, pk_{B_1}, \ldots, pk_{B_n}, pk_{A_0}, pk_{A_1})$
$\sigma^\star \leftarrow \texttt{MDVS.Sign}(\text{params}, m^\star, sk_{A_r}, pk_B)$
$d \leftarrow \mathcal{A}^{\mathcal{H}, \Sigma_0, \Sigma_1, \Upsilon}(\texttt{guess}, \text{params}, m^\star, \mathcal{I}^\star, \sigma^\star, pk_{B_1}, \ldots, pk_{B_n}, pk_{A_0}, pk_{A_1})$
Return d

where $\mathcal{A}$ has access to the oracles $\mathcal{H}$, Σ_0, Σ_1 and Υ. We define the advantage *of the adversary $\mathcal{A}$,* via

$$\mathbf{Adv}^{\mathsf{psi-cma}}_{\textit{MDVS},\mathcal{A}}(k) = \left| Pr\left[\mathbf{Exp}^{\mathsf{psi-cma}-1}_{\textit{MDVS},\mathcal{A}}(k) = 1\right] - Pr\left[\mathbf{Exp}^{\mathsf{psi-cma}-0}_{\textit{MDVS},\mathcal{A}}(k) = 1\right]\right|.$$

MDVS is said to be (k, t, ε)-PSI-CMA secure, if no adversary $\mathcal{A}$ running in time t has an advantage $\mathbf{Adv}^{\mathsf{psi-cma}}_{\textit{MDVS},\mathcal{A}}(k) \geq \varepsilon$.

3 Underlying problems

In this section, we briefly recall the security assumptions upon which are based our bi-designated verifiers signature scheme.

Definition 5 (Admissible bilinear map [4]). *Let $(\mathbb{G}, +)$ and $(\mathbb{H}, \cdot)$ be two groups of the same prime order q and let us denote by P a generator of $\mathbb{G}$.*

An admissible bilinear map *is a map* $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{H}$ *satisfying the following properties:*

- *bilinear:* $e(aQ, bR) = e(Q, R)^{ab}$ *for all* $(Q, R) \in \mathbb{G}^2$ *and all* $(a, b) \in \mathbb{Z}^2$*;*
- *non-degenerate:* $e(P, P) \neq 1$*;*
- *computable: there exists an efficient algorithm to compute* e*.*

Algebraic geometry offers such maps : the Weil and Tate pairings on curves can be used as admissible bilinear maps [4].

Definition 6 (prime-order-BDH-parameter-generator [4]). *A* prime-order-BDH-parameter-generator *is a probabilistic algorithm that takes on input a security parameter* k*, and outputs a 5-tuple* $(q, P, \mathbb{G}, \mathbb{H}, e)$ *satisfying the following conditions:* q *is a prime with* $2^{k-1} < q < 2^k$, $\mathbb{G}$ *and* $\mathbb{H}$ *are groups of order* q*,* P *generates* $\mathbb{G}$*, and* $e : \mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{H}$ *is an admissible bilinear map.*

Now we define the quantitative notion of the complexity of the problems underlying our bi-DVS scheme, namely the Computational Diffie-Hellman Problem (CDH), and the Gap-Bilinear Diffie-Hellman Problem (GBDH).

Definition 7 (CDH). *Let* $\mathcal{G}en$ *be a prime-order-BDH-parameter-generator. Let* D *be an adversary that takes on input a 5-tuple* $(q, P, \mathbb{G}, \mathbb{H}, e)$ *generated by* $\mathcal{G}en$*, and* $(X, Y) \in \mathbb{G}^2$ *and returns an element of* $Z \in \mathbb{G}$*. We consider the following random experiments, where* k *is a security parameter:*

$$\boxed{\textit{Experiment } \mathbf{Exp}^{\mathsf{cdh}}_{\mathcal{G}en,D}(k)}$$

$$\begin{array}{l}
(q, P, \mathbb{G}, \mathbb{H}, e) \xleftarrow{R} \mathcal{G}en(k) \\
\textit{setup} \leftarrow (q, P, \mathbb{G}, \mathbb{H}, e) \\
(x, y) \xleftarrow{R} [\![1, q-1]\!]^2,\ (X, Y) \leftarrow (xP, yP) \\
Z \leftarrow D(\textit{setup}, X, Y) \\
\textit{Return } 1 \textit{ if } Z = xyP,\ 0 \textit{ otherwise}
\end{array}$$

We define the corresponding success *of* D *in solving the CDH problem* via

$$\mathbf{Succ}^{\mathsf{cdh}}_{\mathcal{G}en,D}(k) = Pr\left[\mathbf{Exp}^{\mathsf{cdh}}_{\mathcal{G}en,D}(k) = 1\right]$$

Let $t \in \mathbb{N}$ *and* $\varepsilon \in [0, 1]$*. CDH is said to be* (k, t, ε)*-secure if no adversary* D *running in time* t *has success* $\mathbf{Succ}^{\mathsf{cdh}}_{\mathcal{G}en,D}(k) \geq \varepsilon$*.*

The introduction of bilinear maps in cryptography gives examples of groups where the decisional Diffie-Hellman problem is easy, whereas the computational Diffie-Hellman is still hard. At PKC'01, Okamoto and Pointcheval proposed a new class of computational problems, called *gap problems* [14]. These facts motivated the definition of the following problems:

Computational Bilinear Diffie-Hellman (CBDH): let a, b and c be three integers. Given aP, bP, cP, compute $e(P, P)^{abc}$.

Decisional Bilinear Diffie-Hellman (DBDH): let a, b, c and d be four integers. Given aP, bP, cP and $e(P, P)^d$, decide whether $d = abc \mod q$.

Gap-Bilinear Diffie-Hellman (GBDH): let a, b and c be three integers. Given aP, bP, cP, compute $e(P, P)^{abc}$ with the help of a DBDH Oracle.

Definition 8 (GBDH). *Let $\mathcal{G}en$ be a prime-order-BDH-parameter-generator. Let D be an adversary that takes on input a 5-tuple $(q, P, \mathbb{G}, \mathbb{H}, e)$ generated by $\mathcal{G}en$, and $(X, Y, Z) \in \mathbb{G}^3$ and returns an element of $h \in \mathbb{H}$. We consider the following random experiments, where k is a security parameter:*

$$\boxed{\textit{Experiment } \mathbf{Exp}^{\mathsf{gbdh}}_{\mathcal{G}en,D}(k)}$$

$$\begin{array}{l}
(q, P, \mathbb{G}, \mathbb{H}, e) \xleftarrow{R} \mathcal{G}en(k) \\
\boldsymbol{\mathit{setup}} \leftarrow (q, P, \mathbb{G}, \mathbb{H}, e) \\
(x, y, z) \xleftarrow{R} [\![1, q-1]\!]^3,\ (X, Y, Z) \leftarrow (xP, yP, zP) \\
h \leftarrow D^{\mathcal{O}_{DBDH}}(\boldsymbol{\mathit{setup}}, X, Y, Z) \\
\textit{Return } 1 \textit{ if } h = e(P, P)^{xyz},\ 0 \textit{ otherwise}
\end{array}$$

where $\mathcal{D}^{\mathcal{O}_{DBDH}}$ denotes the fact that the algorithm $\mathcal{D}$ has access to a Decisional Bilinear Diffie-Hellman oracle. We define the corresponding success *of D in solving the GBDH problem* via $\mathbf{Succ}^{\mathsf{gbdh}}_{\mathcal{G}en,D}(k) = Pr\left[\mathbf{Exp}^{\mathsf{gbdh}}_{\mathcal{G}en,D}(k) = 1\right]$.

Let $t \in \mathbb{N}$ and $\varepsilon \in [0, 1]$. GBDH is said to be (k, t, ε)-secure if no adversary D running in time t has success $\mathbf{Succ}^{\mathsf{gbdh}}_{\mathcal{G}en,D}(k) \geq \varepsilon$.

4 Efficient weak-MDVS based on ring signatures.

Efficient construction based on ring signatures Let $\mathcal{R}ing = (\mathtt{Setup}, \mathtt{KeyGen}, \mathtt{Sign}, \mathtt{Verify})$ be a ring signature scheme as defined in [15]. The only requirement concerning this ring signature scheme is that it is "discrete logarithm" based. We mean that the public keys are elements of a unique group $\mathbb{G}$, and the associated private keys their discrete logarithm with respect to a unique generator P. Let $B = \{B_i, i = 1, \ldots, n\}$ be a group of n entities (the designated verifiers), k be an integer and `MDVS` be our new multi-designated verifiers signature scheme with security parameter k.

Setup: $\mathtt{MDVS.Setup} = \mathcal{R}ing.\mathtt{Setup}$
SKeyGen: $\mathtt{MDVS.SKeyGen} = \mathcal{R}ing.\mathtt{KeyGen}$. (P_A, a) is the signer's pair of keys.
VKeyGen: $\mathtt{MDVS.VKeyGen} = \mathcal{R}ing.\mathtt{KeyGen}$. (P_{B_i}, b_i) is a designated verifier's pair of keys, for each $i \in [\![1, n]\!]$.
Sign: A $(B_1, \ldots, B_n)$-designated verifiers signature σ of the message $m \in \{0, 1\}^*$ is produced as follows: $\sigma = \mathcal{R}ing.\mathtt{Sign}\Big(m, P_A, \sum_{i=1}^{n} P_{B_i}, a\Big)$

Verify: $\mathtt{MDVS.Verify}(m, \sigma, P_A, P_{B_1}, \ldots, P_{B_n}) = \mathcal{R}ing.\mathtt{Verify}\Big(m, \sigma, P_A, \sum_{i=1}^{n} P_{B_i}\Big)$

By using a multi-party computation, all the B_i's can cooperate to produce a multi-designated verifier signature corresponding to the public key $P_B = \sum_{i=1}^{n} P_{B_i} = (\sum_{i=1}^{n} b_i)P$. This fact, in addition to the natural property of source hiding of the ring signature, ensures this property for the MDVS scheme.

Security arguments The unforgeability of `MDVS` is guaranted by the unforgeability of the underlying ring signature scheme. The source hiding property comes naturally from the source hiding of the ring signature. The so-built multi-designated verifier signature scheme does not achieve the property of privacy of signer's identity. This can be done by using an encryption layer with an IND-CCA2 cryptosystem (see [13]). Therefore, there is a need for a n-party key agreement protocol for this encryption scheme. In this case of *strong*-MDVS, the protocol looses its spontaneity and becomes less efficient.

5 An efficient and secure strong bi-DVS Scheme

5.1 Description of the scheme B2DVS

We propose an efficient bi-DVS scheme, based on bilinear maps. The efficiency of this scheme comes from the tripartite key exchange based on such maps and described by Joux in [12]. Let us call B and C the designated verifiers. Let $k \in \mathbb{N}$ be the security parameter and $\mathcal{G}en$ be a BDH-prime order generator. Our new scheme B2DVS is designed as follows. It is derived from our previous construction, instanciated with Boneh *et al.*'s ring signatures [5].

Setup: $(q, P, \mathbb{G}, \mathbb{H}, e)$ is the output of $\mathcal{G}en(k)$. Let $[\{0,1\}^* \times \mathbb{H} \longrightarrow \mathbb{G}]$ be a hash function family, and H be a random member of this family
SKeyGen: Alice picks randomly an integer $a \in [\![1, q-1]\!]$ and computes the point $P_A = aP$. Alice's public key is P_A and the secret one is a.
VKeyGen: Bob (resp. Cindy) picks randomly an integer $b \in [\![1, q-1]\!]$ (resp. $c \in [\![1, q-1]\!]$) and computes the point $P_B = bP$ (resp. $P_C = cP$). Bob (resp. Cindy)'s public key is P_B (resp. P_C) and the secret one is b (resp. c)
Sign: Given a message $m \in \{0,1\}^*$, Alice picks at random two integers $(r, \ell) \in [\![1, q-1]\!]^2$, computes $P_{BC} = P_B + P_C$, $u = e(P_B, P_C)^a$, and $M = H(m, u^\ell)$, sets $Q_A = a^{-1}(M - rP_{BC})$ and $Q_{BC} = rP$. The signature σ of m is (Q_A, Q_{BC}, ℓ)
Verify: Given m and σ, Bob (resp. Cindy) computes the value $u = e(P_A, P_C)^b$ (resp. $u = e(P_A, P_B)^c$), and $M = H(m, u^\ell)$. Finally, they test whether $e(Q_A, P_A)e(Q_{BC}, P_{BC}) = e(M, P)$.

Correctness and source hiding of B2DVS are straightforward.

Efficiency considerations Our bi-DVS scheme is very efficient in terms of signature generation, as there are essentially 3 scalar multiplications on a curve and 1 exponentiation in a finite field to perform. The size of the signature is quite short, as it consists in just two points on a curve and some additional random salt. Practically, the signature size is around 480 bits. The computational cost of the verification is essentially the cost of 3 evaluations of the pairing. However this remains very practical, as the computation of algebraic pairings become faster and faster.

5.2 Security Proofs.

The method of our proofs is inspired by Shoup [16]: we define a sequence of games Game_0, Game_1, ... of modified attacks starting from the actual adversary. In each case, all the games operate on the same underlying probability space: the public and private keys of the signature schemes, the coin tosses of the adversary $\mathcal{A}$, the random oracles $\mathcal{H}$.

Theorem 1 (Unforgeability of B2DVS). *Let k be an integer and $\mathcal{A}$ be an EF-CMA-adversary, in the random oracle model, against the bilinear bi-designated verifiers signature scheme B2DVS, with security parameter k, that produces an existential forgery with probability $\varepsilon = \mathbf{Succ}^{\mathsf{ef-cma}}_{B2DVS,\mathcal{A}}(k)$, within time t, making $q_{\mathcal{H}}$ queries to the hash function $\mathcal{H}$ and q_{Σ} queries to the signing oracle.*

Then, there exist $\varepsilon' \in [0,1]$ and $t' \in \mathbb{N}$ verifying $\varepsilon' \geq \left(\frac{1}{2}\varepsilon - \frac{q_{\mathcal{H}}q_{\Sigma}+1}{2^k}\right)^2$ and $t' \leq 2\Big(t + (q_{\mathcal{H}} + 2q_{\Sigma} + O(1))T_M + q_{\Sigma}T_{\mathbb{H}}\Big)$ such that CDH can be solved with probability ε', within time t'. T_M denotes the time complexity to perform a scalar multiplication in $\mathbb{G}$ and $T_{\mathbb{H}}$ the time complexity to perform an exponentiation in $\mathbb{H}$.

Proof. We consider an $\mathsf{EF}-\mathsf{CMA}$-adversary $\mathcal{A}$ outputting an existential forgery $(m^\star, \sigma^\star)$ with probability $\mathbf{Succ}^{\mathsf{ef-cma}}_{\mathsf{B2DVS},\mathcal{A}}(k)$, within time t. We denote by $q_{\mathcal{H}}$ and q_{Σ} the number of queries from the random oracle $\mathcal{H}$ and from the signing oracle Σ. As the attacker can corrupt Bob or Cindy (*i.e.* only one of the two designated verifiers) to obtain their secrets, he knows especially the common key $u = e(P,P)^{abc}$ and therefore can check the validity of the signature by himself.

Let $R_x = xP$, $R_{xy} = xyP$ be two elements in $\mathbb{G}$ for (x,y) in $[\![1, q-1]\!]^2$. We construct a machine which computes the point yP from these points. The CDH problem can be solved by solving two instances of this previous problem (see [5]).

We start by playing the game coming from the actual adversary, and modify it step by step, until we reach a final game whose success probability has an upper bound related to solving this problem. In any Game_j, we denote by $\mathsf{Forge_j}$ the event $B2DVS.\mathtt{Verify}(\mathtt{params}, m, P_A, P_B, P_C, s, \sigma) = 1$ for $s = c$ or $s = b$.

- Game_0 The key generation algorithm for the verifiers is run twice and produces 2 pairs of keys (b, P_B) and (c, P_C) and the key generation algorithm for the signer is run once and produces (a, P_A). The adversary $\mathcal{A}$ is fed with P_A, P_B and P_C, and querying the random oracles $\mathcal{H}$, the signing oracle Σ and, corrupting Bob or Cindy, outputs a pair $(m^\star, \sigma^\star)$. By definition, we have $\Pr[\mathsf{Forge}_0] = \mathbf{Succ}^{\mathsf{ef-cma}}_{\mathsf{B2DVS},\mathcal{A}}(k)$.
- Game_1 We choose randomly an index $i_0 \in \{B, C\}$ and an integer $\alpha \in [\![1, q-1]\!]$. Let $i_1 \in \{B, C\} \setminus \{i_0\}$. We modify the simulation by replacing P_A by R_x, and P_{i_0} by $\alpha R_x - P_{i_1}$ The distribution of (P_A, P_B, P_C) is unchanged since R_x and α are randomly chosen. Therefore $\Pr[\mathsf{Forge}_1] = \Pr[\mathsf{Forge}_0]$.
- Game_2 In this game, we abort if, at any time, the forger corrupts the user i_0. So we have : $\Pr[\mathsf{Forge}_2] = \frac{1}{2}\Pr[\mathsf{Forge}_1]$.

Game_3 In this game, we simulate the random oracle $\mathcal{H}$. For any fresh query $(m, v) \in \{0,1\}^* \times \mathbb{G}$ to the oracle $\mathcal{H}$, we pick $h \in [\![1, q-1]\!]$ at random and compute $M = hR_{xy}$. We store (m, v, h, M) in the H-List and return M as the answer to the oracle call. In the random oracle model, this game is clearly identical to the previous one. Hence, $\Pr[\mathsf{Forge}_3] = \Pr[\mathsf{Forge}_2]$.

Game_4 In this game, we only keep executions which output a valid message/signature $(m, (Q_A, Q_{BC}, \ell))$ such that (m, u^ℓ) has been queried from $\mathcal{H}$. This makes a difference only if (Q_A, Q_{BC}, ℓ) is a valid signature on m, while (m, u^ℓ) has not been queried from $\mathcal{H}$. Since $\mathcal{H}(m, u^\ell)$ is uniformly distributed, the equality $e(Q_A, P_A)e(Q_{BC}, P_{BC}) = e(\mathcal{H}(m, u^\ell), P)$ happens with probability 2^k. Therefore, $|\Pr[\mathsf{Forge}_4] - \Pr[\mathsf{Forge}_3]| \leq 2^{-k}$.

Game_5 Finally, we simulate the signing oracle: for any m, whose signature is queried, we take at random $a_2 \in [\![1, q-1]\!]$ and $(l, r) \in [\![1, q-1]\!]^2$ and set $a_1 = r - a_2\alpha$. If the H-List includes a quadruple $(m, u^l, ?, ?)$ we abort the simulation, otherwise, we store in the H-List the quadruple (m, u^l, r, rR_x) and once we have set $Q_A = a_1 P$ and $Q_{BC} = a_2 P$, (Q_A, Q_{BC}, ℓ) provides a valid signature of m. If it does not abort, this new oracle perfectly simulates the signature. As we abort with probability at most $q_\mathcal{H} 2^{-k}$, we have $|\Pr[\mathsf{Forge}_5] - \Pr[\mathsf{Forge}_4]| \leq q_\mathcal{H} q_\Sigma 2^{-k}$.

At the end of the game 5, the attacker produce a forgery $(m^\star, Q_A^\star, Q_{BC}^\star)$, and by definition of the existential forgery, there is in the H-List a quadruple $(m^\star, v^\star, h^\star, M^\star)$ such that $R_y = {h^\star}^{-1}(Q_A^\star + \alpha Q_{BC}^\star)$ is equal to yP. Thanks to the remark at the beginning of the proof, the success to solve CDH is: $\mathbf{Succ}^{\mathsf{cdh}}_{\mathsf{Game}_5}(k) \geq \left(\frac{1}{2}\mathbf{Adv}^{\mathsf{ef-cma}}_{\mathsf{B2DVS}_B, \mathcal{A}} - \frac{q_\mathcal{H} q_\Sigma + 1}{2^k}\right)^2$. □

Theorem 2 (Privacy of signer's identity in B2DVS). *Let k be an integer and $\mathcal{A}$ be a PSI-CMA-adversary, in the random oracle model, against the bi-designated verifiers signature scheme $B2DVS$, with security parameter k, which has an advantage $\varepsilon = \mathbf{Adv}^{\mathsf{psi-cma}}_{B2DVS, \mathcal{A}}(k)$, within time t, making $q_\mathcal{H}$ queries to the hash function $\mathcal{H}$, q_Σ queries to the signing oracle and q_Υ queries to the verifying oracle. Then, there exist $\varepsilon' \in [0,1]$ and $t' \in \mathbb{N}$ verifying $\varepsilon' \geq \dfrac{\varepsilon}{2} - \dfrac{q_\Upsilon}{2^k} - \dfrac{(q_\mathcal{H} + q_\Sigma)q_\Sigma}{2^k}$ and $t' \leq t + ((q_\mathcal{H} + q_\Sigma)^2 + q_\mathcal{H})(T_{DBDH} + T_\mathbb{H} + O(1)) + q_\Upsilon(3T_P + T_\mathbb{H} + O(1)) + q_\Sigma(4T_\mathbb{G} + O(1))$, such that GBDH can be solved with probability ε', within time t'. T_{DBDH} denotes the time complexity of the DBDH oracle, $T_\mathbb{H}$, $T_\mathbb{G}$, T_P the time complexity to evaluate an exponentiation in $\mathbb{H}$, a scalar multiplication in $\mathbb{G}$ and a pairing.*

Proof. Let $X = xP$, $Y = yP$, $Z = zP$ be a random instance of GBDH. We build a machine computing $u = e(P, P)^{xyz}$ thanks to a DBDH oracle.

Game_0 This is the real attack game, in the random oracle model. We consider a PSI-CMA-adversary $\mathcal{A}$ with advantage $\mathbf{Adv}^{\mathsf{psi-cma}}_{\mathsf{B2DVS}, \mathcal{A}}(k)$, within time t. Two pairs of keys (P_B, b) and (P_C, c) are produced by the key generation algorithm for the verifiers, and two pairs of keys (P_{A_0}, a_0) and (P_{A_1}, a_1) are produced by the key generation algorithm for the signers. $\mathcal{A}$ is fed with the public keys $P_B, P_C, P_{A_0}, P_{A_1}$ and outputs a message $m^\star$ at the end of the `find` stage. Then a signature is performed by flipping a coin $b \in \{0,1\}$ and applying the signing algorithm : $\sigma^\star = B2DVS.\mathtt{Sign}(m^\star, a_b, P_B, P_C)$. This signature is given to $\mathcal{A}$ which outputs a bit $b^\star$ at the end of the `guess` stage. The adversary has a permanent access to the random oracle $\mathcal{H}$, the signing oracles Σ_0 and Σ_1, and the verifying oracle Υ. We denote $q_\mathcal{H}$, q_{Σ_0}, q_{Σ_1} and q_Υ the number of queries to the corresponding oracles. We denote by Guess_0 the event $b^\star = b$, and use a similar notation Guess_i in any Game_i. By definition, we have: $2\Pr[\mathsf{Guess}_0] = 1 + \mathbf{Adv}^{\mathsf{psi-cma}}_{\mathtt{B2DVS},\mathcal{A}}(k)$.

Game_1 We pick at random an integer $\alpha \in [\![1, q-1]\!]$ and modify the simulation by replacing P_{A_0} by X, P_{A_1} by αX, P_B by Y and P_C by Z. The distribution of $(P_{A_0}, P_{A_1}, P_B, P_C)$ is unchanged since (X, Y, Z) is a random instance of the GBDH problem and α is random. Therefore $\Pr[\mathsf{Guess}_1] = \Pr[\mathsf{Guess}_0]$.

Game_2 In this game, we simulate the random oracle $\mathcal{H}$ and maintain an appropriate list, which we denote by $\mathsf{H\text{-}List}$. For any query $(m, v) \in \{0,1\}^* \times \mathbb{H}$

- we check whether the $\mathsf{H\text{-}List}$ contains a quadruple $(m, v, \perp, M)$. If it does, we output M as the answer to the oracle call,
- else we browse the $\mathsf{H\text{-}List}$ and check for all quadruple $(m, \perp, \ell, M)$ whether $(X, Y, Z, v^{1/\ell})$ is a valid Bilinear Diffie-Hellman quadruple. If it does, we output M as the answer to the oracle call,
- otherwise we pick at random $M \in \mathbb{G}$, record $(m, v, \perp, M)$ in the $\mathsf{H\text{-}List}$, and output M as the answer to the oracle call.

In the random oracle model, this game is identical to the previous one. Therefore we get $\Pr[\mathsf{Guess}_2] = \Pr[\mathsf{Guess}_1]$.

Game_3 In this game, we simulate the signing oracles Σ_0 and Σ_1: for any m, whose signature is queried to Σ_i ($i \in \{0,1\}$), by either the adversary or the challenger, we pick at random $(q_A, q_B) \in [\![1, q-1]\!]^2$, $\ell \in [\![1, q-1]\!]$ and computes $M = q_A P_{A_i} + q_B P_B$, $Q_A = q_A \alpha^i P$ and $Q_B = q_B P$.

- If the $\mathsf{H\text{-}List}$ includes a quadruple $(m, \perp, \ell\alpha^i, ?)$, we abort the simulation,
- else we browse the $\mathsf{H\text{-}List}$ and check for each quadruple $(m, v, \perp, ?)$, whether $(X, Y, Z, v^{1/\ell})$ is a valid bilinear Diffie-Hellman quadruple. If it does, we abort the simulation,
- otherwise we add the quadruple $(m, \perp, \ell\alpha^i, M)$ to the $\mathsf{H\text{-}List}$ and output (Q_A, Q_B, ℓ) as the signature of m.

Since, there are at most $q_\mathcal{H} + q_\Sigma$ messages queried to the random oracle $\mathcal{H}$, the new simulation abort with probability at most $(q_\mathcal{H} + q_\Sigma)2^{-k}$. Otherwise, this new oracle perfectly simulates the signature. Summing up for all signing queries, we obtain $|\Pr[\mathsf{Guess}_3] - \Pr[\mathsf{Guess}_2]| \le (q_\mathcal{H} + q_\Sigma) q_\Sigma 2^{-k}$.

Game_4 We simulate the verifying oracle. For any triple message/signature/entity $(m, (Q_A, Q_B, \ell), A_i)$ ($i \in \{0,1\}$), whose verification is queried

- we check whether the $\mathsf{H\text{-}List}$ includes a quadruple $(m, ?, ?, M)$. If it does not, we reject the signature,
- if the $\mathsf{H\text{-}List}$ includes a quadruple $(m, \perp, \ell, M)$, we accept the signature if and only if $e(M, P) = e(Q_A, P_{A_i}) e(Q_B, P_B)$,

- if the H-List includes a quadruple $(m, v, \perp, M)$, we accept the signature if and only if $(X, Y, Z, v^{1/l})$ is a valid bilinear Diffie-Hellman quadruple and $e(M, P) = e(Q_A, P_{A_i})e(Q_B, P_B)$.

This simulation makes a difference only in the first step if (Q_A, Q_B, ℓ) is a valid signature on m, while (m, u^ℓ) has not been queried from $\mathcal{H}$.
Since $\mathcal{H}(m, u^\ell)$ is uniformly distributed, the equality $M = \mathcal{H}(m, u^\ell)$ happens with probability 2^{-k}. Summing up for all verification queries, we get $|\Pr[\mathsf{Guess}_4] - \Pr[\mathsf{Guess}_3]| \leq q_\Upsilon 2^{-k}$.

Game_5 In this game, in the challenge generation, we pick a bit $b \in \{0, 1\}$ at random, and $(Q_A^\star, Q_B^\star, \ell^\star) \in \mathbb{G}^2 \times [\![1, q-1]\!]$, output $(Q_A^\star, Q_B^\star, \ell^\star)$ as the challenge signature of m, but do not update the H-List. This final game is indistinguishable from the previous one unless (m, v) where $v = u^{\ell^\star}$ is queried from $\mathcal{H}$ by the signing oracle, the verifying oracle or the adversary. The first case has already been cancelled in the game Game_3, and by definition of PSI-CMA security, the second case cannot occur, otherwise, the verifying query would be the challenge signature.
The probability that v is queried from $\mathcal{H}$ by the adversary, is upper-bounded by the success ε' to solve the GBDH problem in time t' less than

$$t+((q_\mathcal{H}+q_\Sigma)^2+q_\mathcal{H})(T_{DBDH}+T_\mathbb{H}+O(1))+q_\Upsilon(3T_P+T_\mathbb{H}+O(1))+q_\Sigma(4T_\mathbb{G}+O(1))$$

if we note T_{DBDH} the time complexity of the DBDH oracle, $T_\mathbb{H}$, $T_\mathbb{G}$, T_P the time complexity to evaluate an exponentiation in $\mathbb{H}$, a scalar multiplication in $\mathbb{G}$ and a pairing, since $u = v^{1/\ell^*}$ or $u = v^{1/\alpha\ell^*}$. Thus we get $|\Pr[\mathsf{Guess}_5] - \Pr[\mathsf{Guess}_4]| \leq \varepsilon'$, and since in the game Game_5, the challenge signature gives $\mathcal{A}$ no information about b, we have $\Pr[\mathsf{Guess}_5] = 1/2$.

Summing up the above inequalities, we obtain the claimed bounds. □

Remarks:

- In the simulation, it is possible to abort as soon as the DBDH oracle returns `valid` when requested on a quadruple $(X, Y, Z, v^{1/\ell})$. Indeed, in this case $v^{1/\ell}$ gives the solution to our instance of the GBDH problem.
- Note that in the simulation, we just need a decisional oracle which answers the $(X, Y, Z, \zeta) \in \mathbb{G}^3 \times \mathbb{H}$ instances of the DBDH problem where (X, Y, Z) is fixed. Therefore, the PSI-CMA-security of the scheme can be reduced to a weaker version of the GBDH problem.

6 Final Remarks and Conclusion.

We formally defined the new *multi-designated verifiers signature* primitive (suggested by Desmedt in [8]) and its security model. We proposed an efficient generic construction for weak-DVS schemes, which is based on “discrete-log”-ring signatures. The size of the signatures does not grow with the number of verifiers. Unfortunately, to protect the signer’s anonymity, our scheme needs an additional encryption layer. In the case of two designated verifiers, we proposed a very efficient protocol based on bilinear maps, which achieves the property of privacy of signer's identity without encryption. In general, the encryption layer seems essential to catch the property of privacy of signer's identity , and it is an open problem to build a strong multi-designated verifiers signature scheme without this layer. In a context of RSA signatures [6, 15], this construction can also be used, but besides the encryption layer, the participants B_i's have to generate a shared RSA key in the way of [3] for instance.

References

1. M. Abe, M. Ohkubo, K. Suzuki: 1-out-of-n Signatures from a Variety of Keys. Proc. of Asiacrypt'02, Springer LNCS Vol. 2501, 415–432 (2002)
2. M. Bellare, P. Rogaway: Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. Proc. of 1st ACM Conference on Computer and Communications Security, 62–73 (1993)
3. D. Boneh, and M. Franklin: Efficient generation of shared RSA keys. Journal of the ACM, Vol. 48, Issue 4, 702–722 (2001)
4. D. Boneh, M. Franklin: Identity-based Encryption from the Weil Pairing. SIAM J. Computing, 32 (3), 586–615 (2003).
5. D. Boneh, C. Gentry, B. Lynn, H. Shacham: Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. Proc of Eurocrypt'03, Springer LNCS Vol. 2656, 416–432 (2003)
6. E. Bresson, J. Stern, M. Szydlo: Threshold Ring Signatures for Ad-hoc Groups. Proc. of Crypto'02, Springer LNCS Vol. 2442, 465–480 (2002)
7. D. Chaum: Private Signature and Proof Systems. US Patent 5,493,614 (1996)
8. Y. Desmedt: Verifier-Designated Signatures, Rump Session, Crypto'03 (2003)
9. S. Goldwasser, S. Micali, R. L. Rivest: A digital signature scheme secure against adaptive chosen-message attacks. SIAM J. of Computing, 17 (2) 281–308 (1988)
10. J. Herranz and G. Sáez: Forking Lemmas in the Ring Signatures' Scenario. Proc. of Indocrypt'03, Springer LNCS Vol. 2904, 266–279 (2003)
11. M. Jakobsson, K. Sako, R. Impagliazzo: Designated Verifier Proofs and their Applications. Proc. of Eurocrypt'96, Springer LNCS Vol. 1070, 142–154 (1996)
12. A. Joux: A One Round Protocol for Tripartite Diffie–Hellman. Proc.of ANTS IV, Springer LNCS Vol. 1838, 385–394 (2000)
13. F. Laguillaumie, D. Vergnaud: Designated Verifier Signature: Anonymity and Efficient Construction from *any* Bilinear Map. Proc. of SCN 2004, Springer LNCS, to appear.
14. 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)
15. R. L. Rivest, A. Shamir, Y. Tauman: How to Leak a Secret. Proc. of Asiacrypt'01, Springer LNCS Vol. 2248, 552–565 (2001)
16. V. Shoup: OAEP reconsidered. J. Cryptology, Vol. 15 (4), 223–249 (2002)
17. R. Steinfield, H. Wang, J. Pierprzyk: Efficient Extension of Standard Schnorr/RSA signatures into Universal Designated-Verifier Signatures. Proc. of PKC'04, Springer LNCS Vol. 2947, 86–100 (2004)
18. F. Zhang, R. Safavi-Naini, W. Susilo: An Efficient Signature Scheme from Bilinear Pairings and Its Applications. Proc. of PKC'04, Springer LNCS Vol. 2947, 277–290 (2004)