Efficient Signcryption with Key Privacy from Gap Diffie-Hellman Groups

Benoît Libert Jean-Jacques Quisquater

UCL Crypto Group
Place du Levant, 3. B-1348 Louvain-La-Neuve. Belgium
{libert,jjq}@dice.ucl.ac.be –– http://www.uclcrypto.org/

Abstract. This paper proposes a new public key authenticated encryption (signcryption) scheme based on the Diffie-Hellman problem in Gap Diffie-Hellman groups. This scheme is built on the scheme proposed by Boneh, Lynn and Shacham in 2001 to produce short signatures. The idea is to introduce some randomness into this signature to increase its level of security in the random oracle model and to re-use that randomness to perform encryption. This results in a signcryption protocol that is more efficient than any combination of that signature with an El Gamal like encryption scheme. The new scheme is also shown to satisfy really strong security notions and its strong unforgeability is tightly related to the Diffie-Hellman assumption in Gap Diffie-Hellman groups.

Keywords. signcryption, pairings, Gap Diffie-Hellman groups, provable security

1 Introduction

The concept of public key signcryption schemes was proposed by Zheng in 1997 ([43]). The purpose of this kind of primitive is to perform encryption and signature in a single logical step in order to obtain confidentiality, integrity, authentication and non-repudiation more efficiently than the sign-then-encrypt approach. The drawback of this latter solution is to expand the final ciphertext size (this could be impractical for low bandwidth networks) and increase the sender and receiver's computing time. Several efficient signcryption schemes have been proposed since 1997. The original schemes proposed in [43],[44] were based on the discrete logarithm problem but no security proofs were given. Zheng's original construction was only proven secure in 2002 ([3]) by Baek et al. who described a formal security model in a multi-user setting. In 2000, Steinfeld and Zheng ([41]) proposed another scheme for which the unforgeability of ciphertexts relies on the intractability of the factoring problem but they provided no proof of chosen ciphertext security.

The drawback of the previously cited solutions is that they do not offer easy non-repudiation of ciphertexts: a recipient cannot prove to a third party that some plaintext was actually signcrypted by the sender. Bao and Deng ([5]) proposed a method to add universal verifiability to Zheng's cryptosystem but their scheme was shown ([39]) to leak some information about the plaintext as other schemes described in [42] and [31]. The latter schemes can easily be modified to fix their problem but no strong guarantee of unforgeability can be obtained currently. In the discrete logarithm setting, another scheme was shown in [39] to be chosen ciphertext secure under the Gap Diffie-Hellman assumption. It was a signcryption scheme built on a modified version of the DSA signature scheme which is not provably secure currently. As a consequence, no proof of unforgeability could be found for that scheme. An RSA-based scheme was described by Malone-Lee and Mao ([32]) who provided proofs for both unforgeability under chosen-message attacks and chosen ciphertext security. Unfortunately, they only considered a security in a single-user setting rather than the more realistic multi-user setting. Furthermore, the security of that scheme is only loosely related to the RSA assumption. However, none of these schemes is provably secure against insider attacks: in some of them, an attacker learning some user's private key can recover all messages previously signcrypted by that user while some others are only proven secure against outsider attacks.

In 2002, An et al. ([1]) presented an approach consisting in performing signature and encryption in parallel: a plaintext is first transformed into a pair (c, d) made of a commitment c and a de-commitment d in such a way that c reveals no information about m while the pair (c, d) allows recovering m. Once he completed the transformation, the signer can jointly sign c and encrypt d in parallel using appropriate encryption and signature schemes. The de-signcryption operation is then achieved by the recipient in a parallel fashion: the signature on c is verified while d is decrypted and the pair (c, d) is then used to recover the plaintext. This method decreases the computation time to signcrypt a message to the maximum of the times required by the underlying encryption and signature processes. This parallel approach was further investigated by Pieprzyk and Pointcheval ([35]) who proposed to use a $(2, 2)$-Shamir secret sharing as a commitment scheme: a plaintext is first splitted into two shares s_1, s_2 which do not individually reveal any information on m. s_1 is used as a commitment and signed while s_2 is encrypted as a de-commitment. The authors of [35] also gave a construction allowing them to integrate any one-way encryption system (such as the basic RSA) with a weakly secure signature (non-universally forgeable signatures in fact) into a chosen ciphertext secure and existentially unforgeable signcryption scheme.

Dodis et al. ([20]) recently proposed another technique to perform parallel signcryption. Their method consists in a Feistel probabilistic two-paddings (called PSEP for short) which can be viewed as a generalization of other existing probabilistic paddings (OAEP, OAEP+, PSS-R,etc.) and involve a particular kind of commitment schemes. The authors of [20] showed that their construction also allows optimal exact security, flexible key management, compatibility with PKCS standards and has other interesting properties. On the other hand, all parallel signcryption propositions require the recipient of a message to know from whom a ciphertext emanates before beginning to verify the signature in parallel with the decryption operation. A trivial solution to this problem is to append a tag containing the sender's identity to the ciphertext but this prevents the scheme from satisfying the notion of ciphertext anonymity formalized by Boyen in [14] (intuitively, this notion expresses the inability for someone observing a ciphertext to determine who the sender is nor to whom it is intended) that can be a desirable feature in many applications (see [14] for examples). Furthermore, by the same arguments as those in [7], one can easily notice that the probabilistic padding described in [20] does not allow the key privacy property to be achieved when instantiated with trapdoor permutations such as RSA, Rabin or Paillier: in these cases, given a ciphertext and a set of public keys, it is possible to determine under which key the message was encrypted. An anonymous trapdoor permutation or a repeated variant of the padding PSEP (as the solutions proposed in [7]) could be used to solve this problem but this would decrease the scheme's efficiency.

In this paper, we propose a new discrete logarithm based signcryption scheme which satisfies strong security notions: chosen ciphertext security against insider attacks (except the hybrid composition proposed in [27] and the identity based scheme described in [14], no discrete logarithm based authenticated encryption method was formally proven secure in such a model before), strong unforgeability against chosen-message attacks, ciphertext anonymity in the sense of [14] (this is an extension of the notion of key privacy proposed in [7] to the signcryption case). We also prove that it satisfies a new security notion that is related to the one of ciphertext anonymity and that we call 'key invisibility'. We show that the scheme's strong unforgeability is really tightly related to the hardness of the Diffie-Hellman problem unlike the scheme proposed in [14] whose proof of unforgeability relies on Pointcheval and Stern's forking lemma and thus only provides a loose reduction to a computational problem. In fact, except the hybrid construction of [27] (whose semantic security is based on the stronger hash oracle Diffie-Hellman assumption) our scheme appears to be the first discrete logarithm based signcryption protocol whose (strong) unforgeabililty is proven to be tightly related to the Diffie-Hellman problem. About the semantic security of the scheme, we give heuristic arguments showing that it is more tightly related to the Diffie-Hellman problem than expressed by the bounds at first sight. Unlike [1],[20] and [35], our protocol is sequential but it is efficient and does not require the recipient of a message to know who is the sender before starting the de-signcryption process. Our scheme borrows a construction due to Boyen ([14]) and makes extensive use of the properties of some bilinear maps over the so-called Gap Diffie-Hellman groups (in fact, the structure of these groups is also exploited in our security proofs). Before describing our scheme, we first recall the properties of these maps in section 2. The section 3 formally describes the security notions that our scheme, depicted in section 4, is shown to satisfy in the security analysis presented in section 5.

2 Preliminaries

2.1 Overview of pairings

Let k be a security parameter and q be a $k-$bit prime number. Let us consider groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of the same prime order q. For our purposes, we need a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ satisfying the following properties:

1. Bilinearity: $\forall\ P, Q \in \mathbb{G}_1,\ \forall\ a, b \in \mathbb{Z}_q^*$, we have $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$.
2. Non-degeneracy: for any $P \in \mathbb{G}_1$, $\hat{e}(P, Q) = 1$ for all $Q \in \mathbb{G}_1$ iff $P = \mathcal{O}$.
3. Computability: there is an efficient algorithm to compute $\hat{e}(P, Q)\ \ \forall\ P, Q \in \mathbb{G}_1$.

Modified pairings ([10]) obtained from the Weil or the Tate pairing provide admissible maps of this kind. The group $\mathbb{G}_1$ is a cyclic subgroup of the group of points of a suitable elliptic curve $E(\mathbb{F}_p)$ over a finite field while $\mathbb{G}_2$ is a cyclic subgroup of the multiplicative group associated to a finite extension of $\mathbb{F}_p$. We refer to [33] or [9] for a detailed description of the Weil pairings and to [25] for the Tate pairing. We now recall some problems that provided underlying assumptions for many previously proposed pairing based cryptosystems. These problems are formalized according to the elliptic curve additive notation.

Definition 1. *Given groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order q, a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ and a generator P of $\mathbb{G}_1$,*

- *The **Computational Diffie-Hellman problem** (CDH) in $\mathbb{G}_1$ is, given $\langle P, aP, bP\rangle$ for unknown $a, b \in \mathbb{Z}_q$, to compute $abP \in \mathbb{G}_1$.*
- *The **Decisional Diffie-Hellman problem** (DDH) is, given $\langle P, aP, bP, cP\rangle$ for unknown $a, b, c \in \mathbb{Z}_q$, to decide whether $ab \equiv c \pmod q$ or not. Tuples of the form $\langle P, aP, bP, cP\rangle$ for which the latter condition is verified are called "Diffie-Hellman tuples".*
- *The **Gap Diffie-Hellman problem** (GDH) is to solve a given instance $\langle P, aP, bP\rangle$ of the CDH problem with the help of a DDH oracle that is able to decide whether a tuple $\langle P, a'P, b'P, c'P\rangle$ is such that $c' \equiv a'b' \pmod q$ or not.*

As shown in [28], a pairing can implement a DDH oracle. Indeed, in a group $\mathbb{G}_1$ for which pairings are efficiently computable, to determine whether a tuple $\langle P, aP, bP, cP\rangle$ is a valid Diffie-Hellman tuple or not, it suffices to check if $\hat{e}(P, cP) = \hat{e}(aP, bP)$. This kind of group, where the DDH problem is easy while the CDH one is still believed to be hard, is called Gap Diffie-Hellman groups in the literature ([28],[34]).

3 Security notions for signcryption schemes

We first recall the two usual security notions: the security against chosen ciphertext attacks which is also called semantic security and the unforgeability against chosen-message attacks. We then consider other security notions that were proposed by Boyen ([14]) in 2003. In the notion of chosen ciphertext security, we consider a multi-user security model as already done in [1],[3],[20],[35] and [14] to allow the adversary to query the de-signcryption oracle on ciphertexts created with other private keys than the attacked one. We also consider the security against insider attacks by allowing the attacker to choose to be challenged on a signcrypted text created by a corrupted user (i.e. a user whose private key is known to the attacker). Indeed, for confidentiality purposes, we require the owner of a private key to be unable to find any information on a ciphertext created with that particular key without knowing which randomness was used to produce that ciphertext. This further allows us to show that an attacker stealing a private key does not threaten the confidentiality of messages previously signcrypted using that private key [1].

Definition 2. *We say that a signcryption scheme is semantically secure against chosen ciphertext attacks (we call this security notion SC-IND-CCA) if no probabilistic polynomial time (PPT) adversary has a non-negligible advantage in the following game:*

1. *The challenger runs the key generation algorithm* $\texttt{Keygen}$ *to generate a private/public key pair* (sk_U, pk_U). sk_U *is kept secret while* pk_U *is given to the adversary* $\mathcal{A}$.
2. $\mathcal{A}$ *performs a first series of queries in a first stage. These queries can be of the following kinds:*
   - *Signcryption queries:* $\mathcal{A}$ *produces a message* $m \in \mathcal{M}$ *and an arbitrary public key* pk_R *(that public key may differ from* pk_U*) and requires the result* $\texttt{Signcrypt}(m, sk_U, pk_R)$ *of the signcryption oracle.*
   - *De-signcryption queries:* $\mathcal{A}$ *produces a ciphertext* σ *and requires the result of the operation* $\texttt{De-signcryt}(\sigma, sk_U)$. *This result is made of a signed plaintext and a sender's public key if the obtained signed-plaintext is valid for the recovered sender's public key. Otherwise (that is if the obtained plaintext-signature pair is not valid for the obtained public key when performing the de-signcryption operation with the private key* sk_U*), the* $\perp$ *symbol is returned as a result.*

   These queries can be asked adaptively: each query may depend on the answers to previous ones.
3. $\mathcal{A}$ *produces two plaintexts* $m_0, m_1 \in \mathcal{M}$ *of equal size and an arbitrary private key* sk_S. *The challenger flips a coin* $b \leftarrow_R \{0,1\}$ *to compute a signcryption* $\sigma = \texttt{Signcrypt}(m_b, sk_S, pk_U)$ *of* m_b *with the sender's private key* sk_S *under the attacked receiver's public key* pk_U. σ *is sent to* $\mathcal{A}$ *as a challenge.*
4. *The adversary performs new queries as in the first stage. Now, it may not ask the de-signcryption of the challenge* σ *with the private key* sk_U *of the attacked receiver.*
5. *At the end of the game,* $\mathcal{A}$ *outputs a bit* b' *and wins if* $b' = b$.

$\mathcal{A}$*'s advantage is defined to be* $Adv^{\texttt{ind-cca}}(\mathcal{A}) := 2Pr[b' = b] - 1$.

In the notion of unforgeability captured by the formal definition below, as in many other proposed provably unforgeable signcryption schemes ([1],[3],[14],[20],[27],[35], etc.), we allow a forger attempting to forge a ciphertext on behalf of the attacked user U to know the receiver's private key. In fact, the attacker has to come with the intended receiver's private key sk_R as a part of the forgery. The motivation is to prove that no attacker can forge a ciphertext intended

[1] That feature of signcryption schemes was also considered in [1],[14],[20],[29] and [35] and inappropriately called "forward secrecy" in [29]

to any receiver on behalf of a given sender. In particular, no dishonest user can produce a ciphertext intended to himself and try to convince a third party that it emanates from a honest user.

Definition 3. *We say that a signcryption scheme is strongly existentially unforgeable against chosen-message attacks (SC-SUF-CMA) if no PPT adversary has a non-negligible advantage in the following game:*

1. *The challenger generates a key pair (sk_U, pk_U) and pk_U is given to the forger $\mathcal{F}$.*
2. *The forger $\mathcal{F}$ performs queries to the oracles $\textit{\texttt{Signcrypt}}_{sk_U}(.,.)$ and $\textit{\texttt{De-signcrypt}}_{sk_U}(.)$ exactly as in the previous definition. Again, these queries can also be produced adaptively.*
3. *At the end of the game, $\mathcal{F}$ produces a ciphertext σ and a key pair (sk_R, pk_R) and wins the game if the result of the operation $\textit{\texttt{De-signcrypt}}(\sigma, sk_R)$ is a tuple (m, s, pk_U) such that (m, s) is a valid signature for the public key pk_U such that σ was not the output of a signcryption query $\textit{\texttt{Signcrypt}}(m, sk_U, pk_R)$ made during the game.*

Recall that, in the corresponding notion of conventional (i.e. non-strong) unforgeability for signcryption schemes, the attacker cannot win if the outputted ciphertext was the result of any signcryption query. In our context, as in [1],[27],[20], and many other works, the forger is allowed to have obtained the forged ciphertext as the result of a signcryption query for a different receiver's public key than the one corresponding to the claimed forgery. The only constraint is that, for the message m obtained by de-signcryption of the alleged forgery with the chosen private key sk_R, the outputted ciphertext σ was not obtained as the result of a $\texttt{Signcrypt}(m, sk_U, pk_R)$ query.

In [14], Boyen also proposed additional security notions for signcryption schemes. One of the most important ones was the notion of ciphertext anonymity that can be viewed as an extension to authenticated encryption schemes of the notion of key privacy already considered by Bellare et al in [7]. Intuitively, in the context of public key encryption, a scheme is said to have the key privacy property if ciphertexts convey no information about the public key that was used to create them. In the signcryption setting, we say that the ciphertext anonymity (or key privacy) property is satisfied if ciphertexts contain no information about who created them nor about to whom they are intended. This notion is a transposition into the non-identity based setting of the one presented in [14]. It can be described like that.

Definition 4. *We say that a signcryption scheme satisfies the ciphertext anonymity property (also called key privacy or key indistinguishability: we call this notion SC-INDK-CCA for short) if no PPT distinguisher has a non-negligible advantage in the following game:*

1. *The challenger generates two private/public key pairs $(sk_{R,0}, pk_{R,0})$ and $(sk_{R,1}, pk_{R,1})$. $pk_{R,0}$ and $pk_{R,1}$ are given to the distinguisher $\mathcal{D}$.*
2. *$\mathcal{D}$ adaptively performs queries $\textit{\texttt{Signcrypt}}(m, sk_{R,c}, pk_R)$, for arbitrary recipient keys pk_R, and $\textit{\texttt{De-signcrypt}}(\sigma, sk_{R,c})$ for $c = 0$ or $c = 1$.*
3. *Once stage 2 is over, $\mathcal{D}$ outputs two private keys $sk_{S,0}$ and $sk_{S,1}$ and a plaintext $m \in \mathcal{M}$. The challenger then flips two coins $b, b' \leftarrow_R \{0, 1\}$ and computes a challenge ciphertext $\sigma = \textit{\texttt{Signcrypt}}(m, sk_{S,b}, pk_{R,b'})$ which is sent to $\mathcal{D}$.*
4. *$\mathcal{D}$ adaptively performs new queries as in stage 2 with the restriction that, this time, it is disallowed to ask the de-signcryption of the challenge σ with the private keys $sk_{R,0}$ or $sk_{R,1}$.*
5. *At the end of the game, $\mathcal{D}$ outputs bits d, d' and wins if $(d, d') = (b, b')$. Its advantage is defined to be*

$$Adv^{\textit{\texttt{indk-cca}}}(D) := Pr[(d, d') = (b, b')] - \frac{1}{4}.$$

Again, this notion captures the security against insider attacks since the distinguisher is allowed to choose a set of two private keys among which the one used as sender's key to create the challenge ciphertext is picked by the challenger. The above definition can be viewed as a transposition to the non-identity based setting of the definition of ciphertext anonymity proposed by Boyen ([14]) as well as an extension of the definition of key privacy ([7]) to the authenticated encryption context. We introduce another notion called 'key invisibility' which is close to the concept (formalized by Galbraith and Mao in [23]) of invisibility for undeniable signatures. Intuitively, this notion expresses the impossibility to decide whether a given ciphertext was actually created using a given particular sender's private key and a given particular receiver's public key.

Definition 5. *We say that a signcryption scheme satisfies the key invisibility (we denote this notion by SC-INVK-CCA for short) if no PPT distinguisher has a non-negligible advantage in the following game:*

1. *The challenger generates a private/public key pair* (sk_U, pk_U). pk_U *is given to the distinguisher* $\mathcal{D}$.
2. $\mathcal{D}$ *adaptively performs queries* $\texttt{Signcrypt}(m, sk_U, pk_R)$, *for arbitrary recipient keys* pk_R, *and* $\texttt{De-signcrypt}(\sigma, sk_U)$.
3. *Once stage 2 is over,* $\mathcal{D}$ *outputs a private key* sk_S *and a plaintext* $m \in \mathcal{M}$. *The challenger then flips a coins* $b \leftarrow_R \{0,1\}$. *If* $b = 0$, *then the challenger returns an actual challenge ciphertext* $\sigma = \texttt{Signcrypt}(m, sk_S, pk_U)$ *to* $\mathcal{D}$. *If* $b = 1$, *then the challenger returns a random* σ *uniformly taken from the ciphertext space* $\mathcal{C}$.
4. $\mathcal{D}$ *adaptively performs new queries as in stage 2 with the restriction that, this time, it cannot require the de-signcryption of the challenge* σ *with the private keys* sk_U.
5. *At the end of the game,* $\mathcal{D}$ *outputs bits* d *and wins if* $d = b$. *Its advantage is defined as*

$$Adv^{\texttt{invk-cca}}(D) := 2Pr[d = b] - 1.$$

Again, we allow the distinguisher to choose which private key is used as a part of the challenge to take insider attacks into account.

Galbraith and Mao ([23]) showed that anonymity and invisibility are essentially equivalent security notions for undeniable signatures. While one can prove in the same way that key privacy and key invisibility are also essentially equivalent for some particular encryption schemes, such an equivalence turns out to be unclear in the signcryption case. In fact, one cannot prove that a distinguisher against the key invisibility implies a distinguisher against the key privacy with the same advantage (because two random coins are used by the challenger in the definition of key privacy and a single one for key anonymity). However, we can prove that, for signcryption schemes satisfying some particular properties (that is, for a given message and a given sender's private key, the output of the signcryption algorithm must be uniformly distributed in the ciphertext space when the receiver's public key is random), we can prove that key invisibility implies key privacy. In the next section we propose a scheme that satisfies both of them (in addition to the usual notions of semantic security and unforgeability) in the random oracle model.

4 A Diffie-Hellman based signcryption scheme with key privacy

This section presents a signcryption scheme whose unforgeability under chosen-message attacks is tightly related to the hardness of the computational Diffie-Hellman problem in Gap Diffie-Hellman groups. Our solution relies on the BLS signature (proposed in [11] and recalled in appendix) whose security is enhanced by a random quantity U which is used for encryption purposes but also acts as a random salt to provide a tighter security reduction to the Diffie-Hellman problem in $\mathbb{G}_1$ in the proof of unforgeability.

We assume that both the sender and the receiver agreed on public parameters: security parameters k and ℓ, cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of prime order $q \geq 2^k$ such that ℓ is the number of bits required to represent elements of $\mathbb{G}_1$, a generator P of $\mathbb{G}_1$ and a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$. They also agree on cryptographic hash functions $H_1 : \{0,1\}^{n+2\ell} \rightarrow \mathbb{G}_1$, $H_2 : {\mathbb{G}_1}^3 \rightarrow \{0,1\}^{\ell}$ and $H_3 : \{0,1\}^{\ell} \rightarrow \{0,1\}^{n+\ell}$ where n denotes the size of plaintexts (i.e. the message space is $\mathcal{M} = \{0,1\}^n$). The scheme consists of the following three algorithms (we recall that the symbol $\oplus$ denotes the bitwise exclusive OR).

Keygen: user u picks a random $x_u \leftarrow_R \mathbb{Z}_q$ and sets his public key to $Y_u = x_u P \in \mathbb{G}_1$. His private key is x_u. We will denote the sender and the receiver respectively by $u = S$ and $u = R$ and their key pair by (x_S, Y_S) and (x_R, Y_R).

Signcrypt: to signcrypt a plaintext $m \in \{0,1\}^n$ intended to R, the sender S uses the following procedure

1. Pick a random $r \leftarrow_R \mathbb{Z}_q$ and compute $U = rP \in \mathbb{G}_1$.
2. Compute $V = x_S H_1(m, U, Y_R) \in \mathbb{G}_1$.
3. Compute $W = V \oplus H_2(U, Y_R, rY_R) \in \{0,1\}^{\ell}$ and then scramble the plaintext together with the sender's public key: $Z = (m||Y_S) \oplus H_3(V) \in \{0,1\}^{n+\ell}$.

The ciphertext is given by $\sigma = \langle U, W, Z \rangle \in \mathbb{G}_1 \times \{0,1\}^{n+2\ell}$.

De-signcrypt: when receiving a ciphertext $\sigma = \langle U, W, Z \rangle$, the receiver R has to perform the steps below:

1. Compute $V = W \oplus H_2(U, Y_R, x_R U) \in \{0,1\}^{\ell}$.
2. Compute $(m||Y_S) = Z \oplus H_3(V) \in \{0,1\}^{n+\ell}$. Reject σ if Y_S is not a point on the curve on which $\mathbb{G}_1$ is defined.
3. Compute $H = H_1(m, U, Y_R) \in \mathbb{G}_1$ and then check if $\hat{e}(Y_S, H) = \hat{e}(P, V)$. If this condition does not hold, reject the ciphertext.

The consistency of the scheme is easy to verifiy. To prove to a third party that the sender S actually signed a plaintext m, the receiver just has to forward it m and (U, V, Y_R). The third party can then compute H as in the step 3 of unsigncrypt and perform the signature verification as in the same step 3. We note that, in the signcryption algorithm, the recipient's public key must be hashed together with the pair (m, U) in order to achieve the provable strong unforgeability (as shown in the proof of theorem 2).

As pointed out in [24], in some applications, it is interesting for the origin of a signcrypted text to be publicly verifiable (by firewalls for example). In some other applications, it is undesirable: indeed as explained in [14], in some cases, it is better for a signcrypted text not to convey any information about its sender nor about its intended receiver. This property, called anonymity of ciphertexts, is provided by the above scheme as will be shown in the next section.

From an efficiency point of view, we can easily verifiy that the above scheme is at least as efficient and more compact than any sequential composition of the BLS signature ([11]) with any other Diffie-Hellman based chosen ciphertext secure encryption scheme ([2],[4],[19],[21],[22],[36],[40]): indeed only three scalar multiplications in $\mathbb{G}_1$ are required for the signcryption operation while 1 multiplication and 2 pairings must be performed in the de-signcryption process. A sequential combination of the BLS signature with the encryption scheme proposed in [2] would involve an additional multiplication at decryption. If we take $\ell \approx k \geq 160$ (by working with an appropriate elliptic curve), we see that ciphertexts are about 480 bits longer than plaintexts.

Any combination of the BLS signature with a CCA-secure El Gamal type cryptosystem would result in longer final ciphertexts. With the same choice of parameters, a composition of the BLS signature with the lenth-saving El Gamal encryption scheme ([2]) would result in ciphertexts that would be 640 bits longer than plaintexts.

5 Security Analysis

In this section, we first show that an adversary against the SC-IND-CCA security of the scheme implies a PPT algorithm that can solve the Diffie-Hellman problem in $\mathbb{G}_1$ with high probability. This fact is formalized by the following theorem.

Theorem 1. *In the random oracle model, if an adversary $\mathcal{A}$ has a non-negligible advantage ϵ against the SC-IND-CCA security of the above scheme when running in a time t and performing q_{SC} signcryption queries, q_{DSC} de-signcryption queries and q_{H_i} queries to oracles H_i (for $i = 1, \dots, 4$), then there exists an algorithm $\mathcal{B}$ that can solve the CDH problem in the group $\mathbb{G}_1$ with a probability $\epsilon' \geq \epsilon - q_{H_3} q_{DSC}/2^{2k}$ in a time $t' < t + (4q_{DSC} + 2q_{H_2})t_e$ where t_e denotes the time required for one pairing evaluation.*

Proof. We show how to build an algorithm $\mathcal{B}$ that runs the attacker $\mathcal{A}$ as a subroutine to solve the CDH problem in a polynomial time. Let (aP, bP) be a random instance of the CDH problem in $\mathbb{G}_1$. $\mathcal{B}$ plays the role of $\mathcal{A}$'s challenger in the game of definition 2 and first gives $Y_u = bP \in \mathbb{G}_1$ to $\mathcal{B}$ as a challenge public key. $\mathcal{A}$ then adaptively performs hash queries, signcryption queries and de-signcryption queries as explained in the definition. To handle these queries, $\mathcal{B}$ maintains lists L_1, L_2, L_3 to keep track of the answers given to oracle queries on H_1, H_2 and H_3. Hash queries on H_2 and H_3 are treated in the usual way: $\mathcal{B}$ first checks in the corresponding list if the oracle's value was already defined at the queried point. If it was, $\mathcal{B}$ returns the defined value. Otherwise, it returns an uniformly chosen random element from the appropriate range and updates the corresponding list. When a hash query $H_1(m, U, Y_R)$ is performed, $\mathcal{B}$ first looks if the value of H_1 was previously defined for the input (m, U, Y_R). If it was, the previously defined value is returned. Otherwise, $\mathcal{B}$ picks a random $t \leftarrow_R \mathbb{Z}_q$, returns $tP \in \mathbb{G}_1$ as an answer and inserts the tuple (m, U, Y_R, t) into L_1.

Now, let us see how signcryption and de-signcryption queries are dealt with:

- For a signcryption query on a plaintext m with a recipient's public key Y_R both chosen by the adversary $\mathcal{A}$, $\mathcal{B}$ first picks a random $r \leftarrow_R \mathbb{Z}_q$, computes $U = rP \in \mathbb{G}_1$ and checks if L_1 contains a tuple (m, U, Y_R, t) indicating that $H_1(m, U, Y_R)$ was previously defined to be tP. If no such tuple is found, $\mathcal{B}$ picks a random $t \leftarrow_R \mathbb{Z}_q$ and puts the entry (m, U, Y_R, t) into L_1. $\mathcal{B}$ then computes $V = tY_u = t(bP) \in \mathbb{G}_1$ for the random t chosen or recovered from L_1. The rest follows as in the normal signcryption process: $\mathcal{B}$ computes rY_R (for the Y_R specified by the adversary), runs the H_2 simulation process to obtain $h_2 = H_2(U, Y_R, rY_R)$, and then computes $W = V \oplus h_2$ and $Z = (m||Y_u) \oplus h_3$ where h_3 is obtained by simulation of the H_3 oracle on the input V. (U, W, Z) is then returned as a signcryption of m from the sender of public key Y_u to the recipient of public key Y_R.
- For a de-signcryption query on a ciphertext $\langle U, W, Z \rangle$ chosen by $\mathcal{A}$, $\mathcal{B}$ proceeds as follows: it scans the list L_2, looking for tuples $(U, Y_u, S_i, h_{2,i})$ (with $0 \leq i \leq q_{H_2}$) such that $V_i = h_{2,i} \oplus W$ exists in an entry $(V_i, h_{3,i})$ of L_3 and, for the corresponding elements $h_{3,i}$, $(m_i, Y_{S,i}) = h_{3,i} \oplus Z \in \{0,1\}^{n+\ell}$ is such that there exists an entry $(m_i, U, Y_u, h_{1,i})$ in the list L_1. If no such tuples are found, the $\perp$ symbol is returned to $\mathcal{A}$. Otherwise, elements $(m_i, U, V_i, S_i, h_{1,i})$ satisfying those conditions are kept for future examination. If one of them satisfies both $\hat{e}(P, S_i) = \hat{e}(U, Y_u)$ and $\hat{e}(Y_{S,i}, h_{1,i}) = \hat{e}(P, V_i)$, then $\langle m_i, (U, V_i) \rangle$ is returned as a message-signature pair together with the sender's public key $Y_{S,i}$.

At the end of the first stage, $\mathcal{A}$ outputs two plaintexts m_0 and m_1 together with an arbitrary sender's private key x_S and requires a challenge ciphertext built under the recipient's public key Y_u. $\mathcal{B}$ ignores m_0 and m_1 and randomly picks two binary strings $W \leftarrow_R \{0,1\}^{\ell}$ and $Z \leftarrow_R \{0,1\}^{n+\ell}$. A challenge ciphertext $\sigma = \langle U, W, Z\rangle = \langle aP, W, Z\rangle$ is then sent to $\mathcal{A}$ that then performs a second series of queries at a second stage. These queries are handled by $\mathcal{B}$ as those at the first stage. As done in many other papers in the literature, it is easy to show that $\mathcal{A}$ will not realize that σ is not a valid signcryption for the sender's private key x_S and the public key Y_u unless it asks for the hash value $H_2(aP, bP, abP)$. In that case, the solution of the Diffie-Hellman problem would be inserted in L_2 exactly at that moment and it does not matter if the simulation of $\mathcal{A}$'s view is no longer perfect.

At the end of the game, $\mathcal{A}$ produces a result which is ignored by $\mathcal{B}$. The latter just looks into the list L_2 for tuples of the form $(aP, bP, D_i, .)$. For each of them, $\mathcal{B}$ checks whether $\hat{e}(P, D_i) = \hat{e}(aP, bP)$ and, if this relation holds, stops and outputs D_i as a solution of the CDH problem. If no tuple of this kind satisfies the latter equality, $\mathcal{B}$ stops and outputs "failure".

Now to assess $\mathcal{B}$'s probability of success, let us denote by $\texttt{AskH}_2$ the event that $\mathcal{A}$ asks the hash value of abP during the simulation. As done in several papers in the literature (see [40],[10] or [14]), as long as the simulation of the attack's environment is perfect, the probability for $\texttt{AskH}_2$ to happen is the same as in a real attack (i.e. an attack where $\mathcal{A}$ interacts with real oracles). In a real attack we have

$$\Pr[b = b'] \leq \Pr[b = b' | \neg\texttt{AskH}_2]\Pr[\neg\texttt{AskH}_2] + \Pr[\texttt{AskH}_2] = \frac{1}{2} + \frac{1}{2}\Pr[\texttt{AskH}_2]$$

and then we have $\epsilon = 2\Pr[b = b'] - 1 \leq \Pr[\texttt{AskH}_2]$. Now, the probability that the simulation is not perfect remains to be assessed. The only case where it can happen is when a valid ciphertext is rejected in a de-signcryption query. It is easy to see that for every pair $(V_i, h_{3,i})$ in L_3, there is exactly one pair $(h_{1,i}, h_{2,i})$ of elements in the range of oracles H_1 and H_2 providing a valid ciphertext. The probability to reject a valid ciphertext is thus not greater than $q_{H_3}/2^{2k}$. The bound on $\mathcal{B}$'s computation time derives from the fact that every de-signcryption query requires at most 4 pairing evaluations while the extraction of the solution from L_2 implies to compute at most $2q_{H_2}$ pairings.

□

The above security proof makes use of the pairing's bilinearity to handle de-signcryption queries and thus avoids the use of constructions such as [2], [36], [22], [21] that would increase the ciphertext's length or imply additional computation in the de-signcryption operation. This results in a worst-case bound on algorithm $\mathcal{B}$'s computation time that seems to be quite loose: to extract the solution of the CDH problem, $\mathcal{B}$ might have to compute up to $2q_{H_2}$ pairings if $\mathcal{A}$ only queries oracle H_2 on tuples of the form $(aP, bP, .)$. If we allow up to 2^{60} H_2-queries, this appears to be a loose bound at first sight. But we stress that, heuristically, $\mathcal{A}$ has really no interest in asking hash queries on tuples $(aP, bP, .)$ that are not valid Diffie-Hellman tuples: indeed, since we work with Gap Diffie-Hellman groups, $\mathcal{A}$ is able to decide himself whether a given tuple (aP, bP, cP) is a valid one or not and, since hash functions are indistinguishable from true random functions from its point of view, it can gain no information by querying H_2 on invalid tuples. Furthermore, one can argue that an attacker $\mathcal{A}$ querying H_2 on many invalid tuples without checking the validity of these tuples would have no better strategy to find information about the plaintext than computing the XOR of the ciphertext's W-component with random binary strings. Such a strategy would not be more efficient for $\mathcal{A}$ than an exhaustive search of the solution to the Diffie-Hellman instance embedded in the challenge ciphertext. An attacker having a non-negligible advantage against the semantic security would ask much less than 2^{60} (useless) hash values of invalid Diffie-Hellman tuple. We can thus expect that, at the end of the simulation, L_2 only contains a limited number of entries $(aP, bP, .)$.

We note that the bound on $\mathcal{B}$'s probability of success is tight: if we allow $q_{DSC} \leq 2^{30}$ and $q_{H_3} \leq 2^{60}$, with $k \geq 160$, we obtain $q_{H_3} q_{DSC}/2^{2k} \leq 2^{-230}$ which is a negligible function of the parameter k.

We point out that a pairing can be used here to extract the solution to the Diffie-Hellman problem because there is a checking of the ciphertext's validity at the decryption and ciphertexts submitted to the de-signcryption oracle can thus be rejected if the appropriate H_2 query has not been previously performed. A similar trick cannot be used to prove the security of the El Gamal type encryption, consisting in simply taking the XOR of the plaintext with the hash value of an ephemeral Diffie-Hellman key, without applying any padding to the scheme in such a way that its validity can be checked at decryption.

The following theorem claims the strong unforgeability of the scheme.

Theorem 2. *In the random oracle model, if there exists an adversary $\mathcal{F}$ that has a non-negligible advantage ϵ against the SC-SUF-CMA security of the scheme when running in a time t, making q_{SC} signcryption queries, q_{DSC} de-signcryption queries and at most q_{H_i} queries on oracles H_i (for $i = 1, \ldots, 4$), then there exists an algorithm $\mathcal{B}$ that can solve the Diffie-Hellman problem in $\mathbb{G}_1$ with a probability $\epsilon' > \epsilon - q_{SC} q_{H_1}/2^k - q_{DSC} q_{H_3}/2^{2k}$ in a time $t' < t + 4q_{DSC} t_e$ where t_e denotes the time required for a pairing evaluation.*

Proof. given in appendix. □

This time, we obtain bounds that are explicitly tight. With $k \geq 160$, if we allow $q_{H_3} < 2^{60}$ and $q_{H_1} < 2^{50}$ (recall that hashing onto an elliptic curve may not be considered as an operation of unit cost since it requires some extra computation when compared to hashing onto a finite field, see [6] for details), $q_{DSC} < 2^{30}$ we have $q_{SC} q_{H_1}/2^k < 1/2^{80}$ and we still have $q_{DSC} q_{H_3} < 2^{-230}$. We thus have a negligible degradation of $\mathcal{B}$'s probability of success when compared to the adversary's advantage. The bound on $\mathcal{B}$'s running time is also reasonably tight for $q_{DSC} < 2^{30}$.

We now give a proof in the random oracle model that the scheme satisfies the notion of ciphertext anonymity described in section 3.

Theorem 3. *In the random oracle model, assume there exists a PPT distinguisher $\mathcal{D}$ that has a non-negligible advantage against the SC-INDK-CCA security of the scheme when running in a time t, performing q_{SC} signcryption queries, q_{DSC} de-signcryption queries and q_{H_i} queries to oracle H_i (for $i = 1, \ldots, 4$). Then there exists an algorithm $\mathcal{B}$ that solves the CDH problem with an advantage $\epsilon' > \epsilon - 1/2^{n+\ell-1} - q_{DSC} q_{H_3}/2^{2k}$ when running in a time $t' < t + (4q_{DSC} + 2q_{H_2}) t_e$ where t_e denotes the time required for one pairing evaluation.*

Proof. given in appendix. □

Again, the bound on $\mathcal{B}$'s computation time might seem to be meaningless but, as for the proof of theorem 1, we can argue that a distinguisher performing a great number of H_2 queries on invalid Diffie-Hellman tuples would have no better strategy than an exhaustive search for Diffie-Hellman instances embedded in the challenge-ciphertext. However, if we look at the proofs of semantic security and ciphertext anonymity for the scheme described in [13], although no bound is explicitly given for the running time of solvers for the bilinear Diffie-Hellman problems, these bounds are not tighter than ours. Furthermore, the proof of ciphertext anonymity provided in [13] leads to a significant degradation of the solver's advantage when compared to the distinguisher's one. We note that similar heuristic arguments can be given about solver's running time in [13].

Finally, we close the section related to the security analysis with the following theorem related to the key invisibility.

Theorem 4. *In the random oracle model, if there exists a distinguisher $\mathcal{D}$ having a non-negligible advantage ϵ against the SC-INVK-CCA security of the scheme when running in a time t and performing q_{H_i} queries to oracles H_i, for $i = 1, \ldots, 4$, q_{SC} signcryption queries and q_{DSC} de-signcryption queries, then there exists an algorithm $\mathcal{B}$ that solves the CDH problem with an advantage $\epsilon' > \epsilon - 1/2^{n+\ell-1} - q_{DSC}q_{H_3}/2^{2k}$ in a time $t' < t + (4q_{DSC} + 2q_{H_2})t_e$ where t_e is the time required for a pairing evaluation.*

Proof. given in appendix. □

6 Adding ciphertext unlinkability

This section shows how to add the property of ciphertext unlinkability to the scheme. This is a notion introduced in [14]: signcryption schemes providing that property allow the receiver of a signcrypted text to convince a third party that the sender actually signed the original plaintext but without being able to convince that third party that the sender created the received ciphertext (because the receiver would be able to generate the ciphertext himself from the signature thanks to his private key) and without even disclosing any information that could allow the third party to establish a link between the observed signature and the ciphertext. The public parameters are the same as above but an additional hash function $H_4 : {\mathbb{G}_1}^3 \rightarrow \{0,1\}^\ell$ is needed.

Keygen: user U picks a random $x_u \leftarrow_R \mathbb{Z}_q$ and sets his public key to $Y_u = x_u P \in \mathbb{G}_1$. His private key is x_u. We will denote the sender and the receiver respectively by $u = s$ and $u = r$ and their key pair by (x_S, Y_S) and (x_R, Y_R).

Signcrypt: to signcrypt a plaintext $m \in \{0,1\}^n$ intended to R, the sender S proceeds like this:

1. First compute $\omega = H_4(Y_S, Y_R, x_S Y_R) \in \{0,1\}^\ell$.
2. Pick a random $r \leftarrow_R \mathbb{Z}_q$ and compute $U = rP \in \mathbb{G}_1$ and then $U' = U \oplus \omega \in \{0,1\}^\ell$.
3. Compute $V = x_S H_1(m, U', Y_R) \in \mathbb{G}_1$.
4. Compute $W = V \oplus H_2(U, Y_R, rY_R) \in \{0,1\}^\ell$ and then scramble the plaintext together with the sender's public key: $Z = (m||Y_S) \oplus H_3(V) \in \{0,1\}^{n+\ell}$.

The ciphertext is given by $\sigma = \langle U, W, Z \rangle \in \mathbb{G}_1 \times \{0,1\}^{n+2\ell}$.

De-signcrypt: to de-signcrypt a ciphertext $\sigma = \langle U, W, Z \rangle$, the receiver R follows the procedure below:

1. Compute $V = W \oplus H_2(U, Y_R, x_R U) \in \{0,1\}^\ell$. If the obtained V is not a point on the curve on which $\mathbb{G}_1$ is built, reject the ciphertext σ.
2. Compute $(m||Y_S) = Z \oplus H_3(V) \in \{0,1\}^{n+\ell}$. Reject σ if $Y_S \notin \mathbb{G}_1$.
3. Compute $\omega = H_4(Y_S, Y_R, x_R Y_S) \in \{0,1\}^\ell$ and $U' = U \oplus \omega \in \{0,1\}^\ell$.
4. Compute $H = H_1(m, U', Y_R) \in \mathbb{G}_1$ and then check if $\hat{e}(Y_S, H) = \hat{e}(P, V)$. If this condition does not hold, reject the ciphertext.

In that scheme, we also have the non-repudiation of signatures: the receiver can still forward the de-signcrypted plaintext m and the tuple (U', V, Y_R) that can be checked to be a valid signature on m but, furthermore, as in [14], the receiver R is unable to prove that S actually created the ciphertext σ since his knowledge of x_R allows him to produce the encryption of the signature himself. The above scheme thus provides detachable signatures that are unlinkable to ciphertexts. We note that a single additional multiplication in $\mathbb{G}_1$ is required for signcryption and de-signcryption operations to add this feature to the scheme. This unfortunately implies an additional term on the bound of Diffie-Hellman solver's running time in the security proofs (which are omitted here because of space limitation).

7 Conclusions

This paper proposed a new Diffie-Hellman based signcryption scheme that satisfies strong security requirements. It turns out to be the discrete log based signcryption protocol whose unforgeability is the most tightly related to the Diffie-Hellman problem (except the construction in [27], all other existing ones are built on signatures having security proof relying on the forking lemma or on signatures for which no security proof exists and the CCA-security of [27] relies on stronger assumptions than the present scheme's semantic security). By heuristic arguments, we argued that the reduction from an adaptive chosen ciphertext adversary to a solver for the Diffie-Hellman problem is also efficient. We also introduced a new notion of security called 'key invisibility'. This notion is close to the one of 'key indistinguishability' and is also relevant for public key encryption schemes for which it can be shown to be equivalent to the notion of 'key privacy' (provided, for a given plaintext, the output of the encryption algorithm is indistinguishable from a uniform distribution on the ciphertext space when this algorithm is run with random public keys). We raised the open question of this equivalence in the signcryption case and we show (see the appendix C) that for signcryption schemes satisfying a particular property (namely, for a given message and a given sender's private key, the output of the signcryption algorithm must look uniformly distributed when random public keys are used), 'key invisibility' implies 'key privacy'.

References

1. J.-H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In *Advances in Cryptology - Eurocrypt'02*, LNCS 2332, pp. 83–107. Springer, 2002.
2. J. Baek, B. Lee, and K. Kim. Secure Length-Saving ElGamal Encryption under the Computational Diffie-Hellman Assumption. In *Proceedings of ACISP'00*, LNCS 1841, pp. 49–58. Springer, 2000.
3. J. Baek, R. Steinfeld, and Y. Zheng. Formal Proofs for the Security of Signcryption. In *Proceedings of PKC'02*,LNCS 2274, pp. 80–98. Springer, 2002.
4. J. Baek and Y. Zheng. Simple and Efficient Threshold Cryptosystem from the Gap Diffie-Hellman Group. Available at http://citeseer.nj.nec.com/567030.html.
5. F. Bao and R.-H. Deng. A signcryption scheme with signature directly verifiable by public key. In *Proceedings of PKC'98*, LNCS 1998, pp. 55–59, 1998.
6. P.-S.-L.-M. Barreto and H.-Y. Kim. Fast hashing onto elliptic curves over fields of characteristic 3, 2001. eprint available at http://eprint.iacr.org/2001/098/.
7. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-Privacy in Public-Key Encryption. In *Advances in Cryptology - Asiacrypt'01*, LNCS 2248, pp. 566–582. Springer, 2001.
8. M. Bellare, P. Rogaway, *Random oracles are practical: A paradigm for designing efficient protocols*, Proc. of the 1^{st} ACM Conference on Computer and Communications Security, pp. 62-73, 1993.
9. I. Blake, G. Seroussi, and N. Smart. *Elliptic Curves in Cryptography.* Cambridge University Press, 1999.
10. D. Boneh and M. Franklin. Identity Based Encryption From the Weil Pairing. In *Advances in Cryptology - Proceedings of Crypto'01*, LNCS 2139, pp. 213–229. Springer, 2001.
11. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In *Advances in Cryptology - Proceedings of Asiacrypt'01*, LNCS 2248, pp. 514–532. Springer, 2001.
12. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. Extended version of [11]. Available at http://crypto.stanford.edu/ dabo/pubs.html.
13. X. Boyen. Multipurpose identity-based signcryption : A swiss army knife for identity-based cryptography, 2003. extended version of [14], eprint available at http://eprint.iacr.org/2003/163/.
14. X. Boyen. Multipurpose identity-based signcryption: A swiss army knife for identity-based cryptography. In *Advances in Cryptology (CRYPTO '03)*, LNCS 2729, pp. 382–398. Springer, 2003.
15. D. Chaum and H. van Antwerpen. Undeniable signatures. In *Advances in Cryptology - Crypto'89*, LNCS 435, pp. 212–216, 1989.

16. D. Chaum and T.-P. Pedersen. Wallet Databases with Observers. In *Advances in Cryptology - Crypto'92*, LNCS 740, pp. 89–105, 1992.
17. J.-S. Coron. On the Exact Security of Full Domain Hash. In *Advances in Cryptology - Crypto'00*, LNCS 1880, pp. 229–235, 2000.
18. J.-S. Coron. Optimal Security Proofs for PSS and Other Signature Schemes. In *Advances in Cryptology - Eurocrypt'02*, LNCS 2332, pp. 272–287. Springer, 2002.
19. R. Cramer and V. Shoup. A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In *Advances in Cryptology - Crypto'98*, LNCS 1462, pp. 13–25. Springer, 1998.
20. Y. Dodis, M.-J. Freedman, and S. Walfish. Parallel Signcryption with OAEP, PSS-R and other Feistel Paddings. 2003. Available at http://eprint.iacr.org/2003/043/.
21. E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Advances in Cryptology - Crypto'99*, LNCS 1666, pp. 537–554. Springer, 1999.
22. E. Fujisaki and T. Okamoto. How to Enhance the Security of Public-Key Encryption at Minimum Cost. In *Proceedings of PKC'99*, LNCS 1560, pp. 53–68. Springer, 1999.
23. S. Galbraith and W. Mao. Invisibility and anonymity of undeniable and confirmer signatures. In *Topics in Cryptology - proceedings of CT-RSA 2003*, LNCS 2612, pp. 80-97. Springer, 2003.
24. C. Gamage, J. Leiwo, and Y. Zheng. Encrypted message authentication by firewalls. In *Proceedings of PKC'98*, LNCS 1560, pp. 69–81. Springer, 1998.
25. K.-J. Giuliani. Attacks on the elliptic curve discrete logarithm problem, 1999. Master thesis, University of Waterloo, available at http://citeseer.nj.nec.com/477524.html.
26. E.-J. Goh and S. Jarecki. A signature scheme as secure as the diffie-hellman problem. In *Advances in Cryptology - Eurocrypt'03*, LNCS 2656, pp. 401–415. Springer, 2003.
27. I.-R. Jeong, H.-Y. Jeong, H.-S. Rhee, D.-H. Lee, and I.-L. Jong. Provably secure encrypt-then-sign composition in hybrid signcryption. In *Proceedings of ICISC'02*, LNCS 2587, pp. 16-34. Springer, 2002.
28. A. Joux and K. Nguyen. Separating Decision Diffie-Hellman from Diffie-Hellman in cryptographic groups. In *Journal of Cryptology*, volume 16-Number 4, pp. 239–247. Springer, 2003.
29. H.-Y. Jung, D.-H. Lee, L. J.-I., and C. K.-S. Signcryption Schemes with Forward Secrecy. In *Proceedings of WISA'01*, pp. 403–475, 2001.
30. J. Katz and N. Wang. Efficiency improvements for signature schemes with tight security reductions, Proceedings of the 10^{th} ACM Conference on Computer and Communications Security, pp. 155–164, 2003.
31. J. Malone-Lee. Signcryption with non-repudiation, 2002. Technical report available at http://www.cs.bris.ac.uk/Tools/Reports/Ps/2002-malonelee.pdf.
32. J. Malone-Lee and W. Mao. Two birds one stone: Signcryption using RSA. In *Topics in Cryptology - proceedings of CT-RSA 2003*, LNCS 2612, pp. 211-225. Springer, 2003.
33. A.-J. Menezes. *Elliptic curve public key cryptosystems*. Kluwer Academic Publishers, 1995.
34. T. Okamoto and D. Pointcheval. The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes. In *Proceedings of PKC'01*, LNCS 1992. Springer, 2001.
35. J. Pieprzyk and D. Pointcheval. Parallel Authentication and Public-Key Encryption. In *Proceedings of ACISP'03*, LNCS 2727, pp. 383–401. Springer, 2003.
36. D. Pointcheval. Chosen-Ciphertext Security for any One-Way Cryptosystem . In *Proceedings of PKC'00*, LNCS 1751, pp. 129–146. Springer, 2000.
37. D. Pointcheval and J. Stern. Security proofs for signature schemes. In *Advances in Cryptology - Eurocrypt'96*, LNCS 1992, pp. 387–398. Springer, 1996.
38. D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. In *Journal of Cryptology*, volume 13-Number 3, pp. 361–396. Springer, 2000.
39. J.-B. Shin, K. Lee, and K. Shim. New DSA-verifiable signcryption schemes. In *Proceedings of ICISC'02*, LNCS 2587, pp. 35–47. Springer, 2002.
40. V. Shoup and R. Gennaro. Securing Threshold Cryptosystems against Chosen Ciphertext Attack. In *Advances in Cryptology - Eurocrypt'98*, LNCS 1403, pp. 1–16. Springer, 1998.
41. R. Steinfeld and Y. Zheng. A signcryption scheme based on integer factorization. In *Proceedings of ISW'00*, LNCS 1975, pp. 308–322, Springer, 2000.
42. B.-H. Yum and P.-J. Lee. New signcryption schemes based on KCDSA. In *Proceedings of ICISC'01*, LNCS 2288, pp. 305–317, Springer, 2001.
43. Y. Zheng. Digital signcryption or how to achieve cost (signature & encryption) << cost(signature) + cost(encryption). In *Advances in Cryptology - Crypto'97*, LNCS 1294, pp. 165–179, Springer, 1997.
44. Y. Zheng and H. Imai. Efficient signcryption schemes on elliptic curves, 1998. Proc. of IFIP SEC'98, Chapman & Hall.

Appendix A

The BLS signature

In this part of the appendix, we recall the scheme proposed by Boneh et al. ([11]) in 2001 to provide 160-bit signatures using the properties of Gap Diffie-Hellman groups. This signature scheme was previously implicitly suggested in [15],[16] and in [34]. The system makes use of a pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$, where $\mathbb{G}_1$ is of prime order q and has a generator P, and a hash function $h : \{0,1\}^* \rightarrow \mathbb{G}_1$.

Keygen: pick a random $x \leftarrow_R \mathbb{Z}_q$ and compute the public key $Y = xP \in \mathbb{G}_1$. x is the signer's private key.
Sign: the signature on a message M is given by $\sigma = xh(M) \in \mathbb{G}_1$.

Verify: to verify a signature σ on a message M, compute $H = h(M) \in \mathbb{G}_1$ and check if (P, Y, H, σ) is a valid Diffie-Hellman tuple. To do that, it sufficies to ensure that

$$\hat{e}(P, \sigma) = \hat{e}(Y, H).$$

The signature is rejected if this condition is not satisfied.

To obtain short signatures, Boneh et al. propose to use suitable families of elliptic curves and to represent a signature by the $x-$coordinate of the point together with an additional bit. Similarly to the Full Domain Hash signature ([17]), the above one presents a security degradation for the probability of success of an algorithm solving the computational Diffie-Hellman problem when compared to an attacker's advantage in a chosen-message attack against the scheme. However, as noticed in [12], introducing a random salt into the original BLS signature (similarly to the Probabilistic Full Domain Hash scheme depicted in [18]) increases its length but provides it with a security that is quite tightly related to the Diffie-Hellman problem. By an analysis similar to Coron's one ([18]), one can show that a 30-bit random salt is sufficient to enhance the hardness of forging a signature to the same level of difficulty as solving the CDH problem. Even though a technique due to Katz and Wang ([30]) allows removing the random salt without decreasing the security of the scheme, the randomized signature is still of interest since it provides a high level of security at the price of a 30-bit increasing in the length signatures. Recall that, except for the schemes recently proposed in [26] and [30], all discrete logarithm based signatures either have no security proof or have a reduction from a hard computational problem obtained via the Pointcheval-Stern Forking Lemma ([37],[38]). As pointed out in [26] and [30], all security proofs relying on the latter technique present a degradation of security in the reduction.

Appendix B: security proofs

Proof of theorem 2

$\mathcal{B}$ receives a random instance (aP, bP) of the Diffie-Hellman problem. It uses $\mathcal{F}$ as a subroutine to solve that instance and plays the role of $\mathcal{F}$'s challenger in the game of definition 3. It initializes $\mathcal{F}$ with $Y_u = bP$ as a challenge public key. $\mathcal{F}$ then performs adaptive queries that are handled like explained below (using lists L_1, L_2, L_3 as in the proof of theorem 1):

- H_2 and H_3 queries: are dealt with in the usual way as in the proof of theorem 1.
- H_1 queries: when $\mathcal{F}$ asks the hash value of a tuple (m, U, Y_R) that was previously queried, $\mathcal{B}$ returns the value defined at the first query. For a query on a new tuple (m, U, Y_R), $\mathcal{B}$ picks a random $t \in \mathbb{Z}_q$ and defines the value of $H_1(m, U, Y_R)$ to be $t(aP) \in \mathbb{G}_1$ which is returned to $\mathcal{F}$ (the list L_1 is updated accordingly).

- Signcryption queries: for a signcryption query on a message and a receiver's public key Y_R chosen by $\mathcal{F}$, $\mathcal{B}$ picks a random $r \in \mathbb{Z}_q$, computes $U = rP \in \mathbb{G}_1$. If the value of H_1 is already defined at (m, U, Y_R), then $\mathcal{B}$ outputs "failure" and halts. Otherwise, $\mathcal{B}$ picks a random $t \in \mathbb{Z}_q$ and sets $H_1(m, U, Y_R) = tP$ ($\mathcal{B}$ updates L_1 accordingly to be able to answer subsequent hash queries on the input (m, U, Y_R)). $\mathcal{B}$ then computes $V = tY_S$, $W = V \oplus H_2(U, Y_R, rY_R)$ and $Z = (m||Y_S) \oplus H_3(V)$ (where the values of H_2 and H_3 are obtained from oracle simulation algorithms). The ciphertext (U, W, Z) is then returned to $\mathcal{F}$ as a signcryption of m.
- De-signcryption queries: are handled exactly as in the proof of theorem 1.

At the end of the game, $\mathcal{F}$ produces a ciphertext $\langle U', W', Z' \rangle$ and a recipient's public/private key pair (x_R, Y_R). At that moment, $\mathcal{B}$ can perform the de-signcryption using x_R and, if the ciphertext is actually a valid forged ciphertext for the sender's public key Y_u, $\mathcal{B}$ can recover the message-key pair (m', Y_u) and the signature V'. If the hash value $H_1(m', U', Y_R)$ was not asked by $\mathcal{F}$ during the simulation, $\mathcal{B}$ outputs "failure" and stops. Otherwise, the hash value $H_1(m', U', Y_R)$ must have been defined to be $t(aP)$, for some $t \in \mathbb{Z}_q$ which is known to $\mathcal{B}$, and V' must be equal to $t(abP)$ that can then easily extract the solution $t^{-1}V'$ of the CDH problem in $\mathbb{G}_1$. We can now assess $\mathcal{B}$'s probability of success. It is easy to see that the probability for $\mathcal{B}$ to fail in answering a signcryption query is not greater than $q_{SC}q_{H_1}/2^k$ (since at each signcryption query, there is at most q_{H_1} elements in L_1 and the randomly chosen $r \in \mathbb{Z}_q$ is uniformly taken from a set of 2^k elements). As in the proof of theorem 1, the probability to reject a valid ciphertext is not greater than $q_{DSC}q_{H_3}/2^{2k}$. It is easy to see that the probability that $\mathcal{F}$ succeeds in producing a forged ciphertext (U', W', Z') without asking the query $H_1(m', U', Y_R)$ recovered plaintext, is at most $1/2^k$.

□

Proof of theorem 3

Let (aP, bP) be a random instance of the CDH problem. $\mathcal{B}$ uses $\mathcal{A}$ to solve that instance and plays the role of $\mathcal{D}$'s challenger in the game of definition 4. To do this, $\mathcal{B}$ picks random elements $x, y \in \mathbb{Z}_q$ and initializes $\mathcal{D}$ with $pk_{R,0} = Y_{R,0} = x(bP) \in \mathbb{G}_1$ and $pk_{R,1} = Y_{R,1} = y(bP) \in \mathbb{G}_1$. $\mathcal{D}$ then performs queries as explained in definition 4. To deal with these queries, $\mathcal{B}$ maintains lists L_i, for $i = 1, \dots, 3$ to keep track of answers to queries on random oracles H_i. When oracles H_2, H_3 are queried, $\mathcal{B}$ answers in the usual way by returning the previously defined value for the queried input if such a value exists and by returning a new uniformly chosen random value from the appropriate range when the queried oracle has not been defined yet at the queried input. Let us see how other queries are tackled with:

- H_1 queries on an input (m, U, Y_R) are handled as in the proof of theorem 1 by picking a random $t \leftarrow_R \mathbb{Z}_q$, inserting (m, U, Y_R, t) into L_1 and returning $H_1(m, U, Y_R) = tP \in \mathbb{G}_1$.
- Signcryption queries: are treated exactly as in the proof of theorem 1. As in the latter, $\mathcal{B}$ never reaches a state of failure when handling a signcryption query.
- De-signcryption queries: are also treated as in the proof of theorem 1. The probability for $\mathcal{B}$ to reject a valid ciphertext is also bounded by $q_{DSC}q_{H_3}/2^{2k}$.

Once the first stage is finished, $\mathcal{D}$ outputs two sender's private key $sk_{S,0} = x_{S,0} \in \mathbb{Z}_q$, $sk_{S,1} = x_{S,1} \in \mathbb{Z}_q$ and a plaintext $m \in \{0,1\}^n$. $\mathcal{B}$ ignores these outputs and sends the fake ciphertext $\sigma = \langle aP, W, Z \rangle$, where W and Z are randomly chosen binary strings of length ℓ and $n + \ell$, as a challenge to $\mathcal{D}$. It is clear that, as long as $\mathcal{D}$ does not query H_2 to obtain the hash value of $(m, aP, x(bP), x(abP))$ or $(m, aP, y(bP), y(abP))$, it will not realize that σ is a fake ciphertext and the simulation will remain perfect. Moreover, in either of these two events, $\mathcal{B}$ can easily extract the solution abP of the CDH problem thanks to its knowledge of x and y and by computing two pairings.

At the second stage, $\mathcal{D}$ performs queries that are handled exactly as in the first stage. Eventually, $\mathcal{D}$ outputs a guess (d, d') which is ignored by $\mathcal{B}$ that can extract the solution by itself. Suppose that the answer to a H_3 query is never $Z \oplus (m, Y_{S,0})$ nor $Z \oplus (m, Y_{S,1})$ where Z is the random string embedded into the challenge σ, $Y_{S,0} = x_{S,0}P$ and $Y_{S,1} = x_{S,1}P$ (we denote by $\mathtt{BadH}_3$ this 'bad' event that might influence $\mathcal{D}$'s behaviour). Since $\mathcal{D}$ is assumed to have a non-negligible advantage in the real environment of attack, if $\mathtt{BadH}_3$ does not occur, with high probability, it queries H_2 on $(m, aP, x(bP), x(abP))$ or on $(m, aP, x(bP), y(abP))$ during the simulation. Indeed, let $\mathtt{AskH}_2$ denote this event. In the real attack, we have

$$\begin{aligned}
\Pr[(d,d') = (b,b')] &\leq \Pr[(d,d') = (b,b')|\neg\mathtt{BadH}_3 \wedge \neg\mathtt{AskH}_2]\Pr[\neg\mathtt{BadH}_3 \wedge \neg\mathtt{AskH}_2] \\
&\quad + \Pr[(d,d') = (b,b') \wedge \neg\mathtt{AskH}_2|\mathtt{BadH}_3]\Pr[\mathtt{BadH}_3] + \Pr[\mathtt{AskH}_2] \\
&\quad \leq \Pr[(d,d') = (b,b')|\neg\mathtt{BadH}_3 \wedge \neg\mathtt{AskH}_2]\Pr[\neg\mathtt{AskH}_2] + \Pr[\mathtt{BadH}_3] + \Pr[\mathtt{AskH}_2] \\
&\quad = \frac{1}{4} + \frac{2}{2^{n+\ell}} + \frac{3}{4}\Pr[\mathtt{AskH}_2]
\end{aligned}$$

since, if $\mathtt{AskH}_2$ and $\mathtt{BadH}_3$ do not occur during the game, $\mathcal{D}$ has absolutely no information on V (the XOR of W with the appropriate hash value) nor on which sender's key is scrambled together with the plaintext. We then have $\epsilon - 1/2^{n+\ell-1} \leq \Pr[\mathtt{AskH}_2]$. As in theorem 1, one can easily show that, as long as the simulation is perfect (i.e. as long as a valid ciphertext is not rejected), the probability for $\mathtt{AskH}_2$ to happen is the same in the simulation as in a real attack. $\mathcal{B}$'s probability of success is thus at least $\epsilon - q_{DSC}q_{SC}/2^{2k}$. Its strategy to solve the CDH problem is simply to compute two pairings for each H_2 query made by $\mathcal{D}$ on tuples $(aP, x(bP), .)$ or $(aP, y(bP), .)$. This gives us the same bound on $\mathcal{B}$'s running time as the one of theorem 1.

□

Proof of theorem 4

Let (aP, bP) be a random instance of the CDH problem given to $\mathcal{B}$. The distinguisher $\mathcal{D}$ is initialized with the public key $Y_u = bP$. It performs queries that are all handled as in the proof of theorem 3. At some time, $\mathcal{D}$ outputs a plaintext $m \in \{0,1\}^n$ together with a private key $x_S \in \mathbb{Z}_q$ and requests a challenge from $\mathcal{B}$. $\mathcal{B}$ then picks random binary strings $W \leftarrow_R \{0,1\}^\ell$, $Z \leftarrow \{0,1\}^{n+\ell}$ and sends $\sigma = \langle aP, W, Z\rangle$ as a challenge to $\mathcal{D}$. As in the proof of theorem 3, $\mathcal{D}$ will not realize that σ is actually a fake ciphertext until it asks the hash value $H_2(aP, bP, abP)$. As explained in the previous proofs, this event is very likely to happen but, since it provides $\mathcal{B}$ with the solution to its problem, it does not matter if the simulation is no longer perfect. Now, we can assess the probability $\mathtt{AskH}_2$ that the solution to the CDH problem is asked to oracle H_2. Let $\mathtt{BadH}_3$ be the 'bad' event that $Z \oplus (m, Y_S)$ is the answer of a H_3 query at some time, where $Y_S = x_SP$, (we call this a 'bad' event since it might influence $\mathcal{D}$'s behaviour). Since $\mathcal{D}$ is assumed to have a non-negligible advantage in the real environment of attack, if $\mathtt{BadH}_3$ does not occur, with high probability, it queries H_2 on (m, aP, bP, abP) during the simulation. In the real attack, we have

$$\begin{aligned}
\Pr[d = b] &\leq \Pr[d = b|\neg\mathtt{BadH}_3 \wedge \neg\mathtt{AskH}_2]\Pr[\neg\mathtt{BadH}_3 \wedge \neg\mathtt{AskH}_2] \\
&\quad + \Pr[d = b \wedge \neg\mathtt{AskH}_2|\mathtt{BadH}_3]\Pr[\mathtt{BadH}_3] + \Pr[\mathtt{AskH}_2] \\
&\quad \leq \Pr[d = b|\neg\mathtt{BadH}_3 \wedge \neg\mathtt{AskH}_2]\Pr[\neg\mathtt{AskH}_2] + \Pr[\mathtt{BadH}_3] + \Pr[\mathtt{AskH}_2] \\
&\quad = \frac{1}{2} + \frac{1}{2^{n+\ell}} + \frac{1}{2}\Pr[\mathtt{AskH}_2]
\end{aligned}$$

and then $\Pr[\mathtt{AskH}_2] \geq \epsilon - 1/2^{n+\ell-1}$. Since $\mathcal{B}$ can find the solution to the CDH instance as soon as $\mathtt{AskH}_2$ happens, its probability of success is simply for $\mathtt{AskH}_2$ to happen decreased by the probability for the simulation not to be perfect (i.e. for a valid ciphertext to be rejected), we obtain the same bounds on $\mathcal{B}$'s advantage and on its running time as in theorem 3.

□

Appendix C: proof of the relation between key invisibility and key privacy

We consider a signcryption scheme (K, SC, DSC) made of a key generation algorithm K, a signcryption algorithm SC and a de-signcryption algorithm DSC. We assume the scheme has the following property: for any given m in the space of plaintexts $\mathcal{M}$ and any sk in the space of private keys $\mathcal{SK}$, the distribution $\{\sigma \in \mathcal{C} | \sigma = SC(m, sk, pk) \wedge pk \leftarrow_R \mathcal{PK}\}$ is indistinguishable from a uniform distribution on the space of ciphertexts $\mathcal{C}$, where $\mathcal{PK}$ denotes the space of public keys.

Now we consider, a disinguisher D_A against the key privacy of such a scheme. We show that it implies the existence of a distinguisher D_I against its key invisibility. The proof is inspired from the one of Galbraith and Mao ([23]) for undeniable signatures. The distinguisher, depicted below, takes as input a public key pk_0 and has access to a challenger $Chall$ such as the one of definition 5.

```
$D_I^{Chall}(pk_0)$
  1. $(pk_1, sk_1) \leftarrow K(k)$
  2. $b' \leftarrow_R \{0,1\}$
     If $b' = 0$ send $(pk_0, pk_1)$ to $D_A$
     else send it $(pk_1, pk_0)$.
  3. For any query related to the private key
     associated to $pk_0$, forward it to $Chall$ and
     transmit the answer to $D_A$. Answer any query
     related to $pk_1$ with the private key $sk_1$.
  4. For the tuple $(m, sk'_0, sk'_1)$ outputted by $D_A$,
     flip a coin $b'' \leftarrow_R \{0,1\}$; send $(m, sk'_{b''})$ to $Chall$.
  5. Receive a challenge $\sigma$ from $Chall$
     and forward it to $D_A$.
  6. Handle all queries made by $D_A$ as in step 3.
  7. $D_A$ outputs $(d', d'')$. Return 0 if $(d', d'') = (b', b'')$
     and 1 otherwise.
```

Let b be the random bit chosen by D_I's challenger in the game of definition 5 and let ϵ be D_A's advantage in the game of definition 4. From D_I's behaviour, it is clear that, if $b = 0$ (that is if $Chall$ decides to return an actual ciphertext for the private key $sk'_{b''}$ and the public key pk_0), we have $\Pr[(d', d'') = (b', b'')] = \epsilon + 1/4$. Now, if $b = 1$ (that is if $Chall$ decides to return a randomly chosen σ from the ciphertext space), it is clear that, with overwhelming probability, σ is not a valid ciphertext that could be the output of $SC(m, sk'_u, pk_v)$ for $u, v = 0, 1$. Nevertheless, because of the aforementioned assumed property of the scheme, D_A is unable to realize it and it comes that $\Pr[(d', d'') = (b', b'') | b = 1] \leq 1/4$ and we can then write

$$\begin{aligned}\Pr[\mathtt{Output}(D_I) = b] &= \Pr[(d', d'') = (b', b'') | b = 0]\Pr[b = 0] + \Pr[(d', d'') \neq (b', b'') | b = 1]\Pr[b = 1] \\ &\geq (\epsilon + \frac{1}{4})\frac{1}{2} + \frac{3}{4}\frac{1}{2} = \frac{\epsilon}{2} + \frac{1}{2}\end{aligned}$$

and hence, $\mathtt{Adv}(D_I) = 2\ \Pr[\mathtt{Output}(D_I) = b] - 1 \geq \epsilon$.

We find that, for a signcryption scheme satisfying the required property, a distinguisher against the key invisibility has an advantage at least as large as the one of a distinguisher against the key privacy. This is not so surprising since, for our scheme, we obtain similar bounds for the reductions of theorems 3 et 4.