Improved Signcryption from q-Diffie-Hellman Problems

Benoît Libert[1,2] * Jean-Jacques Quisquater [1]

[1]UCL Crypto Group
Place du Levant, 3. B-1348 Louvain-La-Neuve. Belgium
{libert,jjq}@dice.ucl.ac.be
[2] Laboratoire d'Informatique de l'École Polytechnique (LIX)
F-91128 Palaiseau CEDEX, France

Abstract. This paper proposes a new public key authenticated encryption (signcryption) scheme based on the hardness of q-Diffie-Hellman problems in Gap Diffie-Hellman groups. This new scheme is quite efficient: the signcryption operation has almost the same cost as an El Gamal encryption while the reverse operation only requires one pairing evaluation and three exponentiations. The scheme's chosen-ciphertext security is shown to be related to the hardness of the q-Diffie-Hellman Inversion ($q-$DHI) problem in the random oracle model while its unforgeability is proved under the q-Strong Diffie-Hellman assumption (q-SDH). It also provides detachable signatures that are unlinkable to the original anonymous ciphertext. We also show that most of the sender's workload can be computed offline. Our construction is based on a signature scheme independently studied by Boneh-Boyen and Zhang et al. in 2004.

Keywords. signcryption, bilinear maps, provable security.

1 Introduction

Public key encryption and digital signatures are primitives that aim at achieving very different purposes. The former has to yield communication privacy while the latter is used for authentication and non-repudiation of sensitive data. However, in many cryptographic applications, such as secure e-mail or secure channel establishment protocols, these requirements need to be simultaneously fulfilled. To achieve them in the asymmetric setting, the concept of public key authenticated encryption, or 'signcryption', was introduced by Zheng in 1997 [36]. This kind of primitive aims to efficiently perform encryption and signature in a single logical step in order to obtain confidentiality, integrity, authentication and non-repudiation. We recall that the basic encrypt-then-sign composition is generally insecure (except for some particular constructions such as [1] or [16] or if, as in [2], we consider a relaxed security notion for this kind of composition) against chosen-ciphertext attacks as well as the encrypt-and-sign approach. The drawback of most of the latter solutions is to expand the final ciphertext size

* This author was supported by the DGTRE's First Europe Program

and increase the sender and receiver's computing time. Zheng's original discrete logarithm based signcryption proposal [36] was only proven secure in 2002 [3] by Baek et al. who described a formal security model in a multi-user setting. In [32], Steinfeld and Zheng proposed another scheme for which the unforgeability of ciphertexts relies on the intractability of the integer factoring problem but they provided no proof of chosen-ciphertext security.

The previously cited schemes have the shortcoming not to offer easy non-repudiation of ciphertexts: a recipient cannot convince a third party that a plaintext actually emanates from its sender. A method was proposed in [4] to overcome this limitation but it was shown [30] to leak information on the plaintext as other schemes described in [21] and [34]. This weakness can easily be fixed by slightly modifying the schemes but no strong guarantee of unforgeability can be obtained since only loose reductions from computational problems to a ciphertext forger currently exist. Formal security notions were considered in [1] where general composition methods for asymmetric encryption and digital signatures together are analyzed but these security models do not consider that, in a multi-user setting, an adversary can be an insider as stressed in [2]: a chosen-ciphertext attacker can learn some user's private key and threaten the confidentiality of messages previously signcrypted by that user.

Another CCA2-secure discrete logarithm based signcryption scheme was described in [30] but no proof of unforgeability was given for it. An RSA-based scheme was described by Malone-Lee and Mao [22] who provided proofs for unforgeability under chosen-message attacks and chosen-ciphertext security but they only considered the security in a two-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 the aforementioned schemes is provably secure against insider attacks: in some of them, an attacker learning some user's private key can recover all the messages previously sent by that user.

An et al. [2] presented an approach consisting in performing signature and encryption in parallel: a plaintext is first transformed into a commitment/decommitment pair (c, d) in such a way that c reveals no information about m while the pair (c, d) allows recovering m. The signer can then jointly sign c and encrypt d in parallel using appropriate encryption and signature schemes. The de-signcryption operation is also achieved by the recipient in a parallel fashion: the signature on c is verified while the decryption reveals d and the pair (c, d) is finally used to recover the plaintext. This approach was further investigated by Pieprzyk and Pointcheval [28] who proposed to use a commitment scheme based on a $(2, 2)$-Shamir secret sharing of an appropriately salted plaintext: the first resulting share s_1, which does not individually reveal any information on m, is used as a commitment and is signed while the second share s_2 is encrypted as a de-commitment. That method also provides a construction allowing to integrate any one-way encryption system (such as the basic RSA) with a weakly secure signature (a non-universally forgeable signatures) into a CCA2-secure and existentially unforgeable scheme.

Dodis et al. [12] recently proposed other possibly parallel signcryption techniques, one of which which can be viewed as a generalization of existing probabilistic paddings such as OAEP, OAEP+ or PSS-R. They showed that their constructions allow optimal exact security as well as compatibility with PKCS standards and has other interesting properties. In fact, the latter schemes might be the most practical ones among all solutions based on trapdoor functions.

In a discrete logarithm context, Libert and Quisquater recently proposed a solution based on bilinear maps and the properties of Gap Diffie-Hellman groups. This scheme was shown to satisfy strong security notions (namely, notions of chosen-ciphertext security and 'ciphertext strong unforgeability' even against insider attacks that were introduced in [2] where the multi-user setting was considered for the first time) and to provide anonymous ciphertexts (i.e. a ciphertext contains no information identifying its originator nor its recipient) as well as a notion called 'invisibility' (that informally captures the fact that an actual ciphertext for a given recipient's public key and a chosen sender's private key is indistinguishable from a random element of the ciphertext space). Unfortunately, the latter Diffie-Hellman based scheme, that was obtained from Boneh et al.'s short signature [10] in a randomized version, only spares one elliptic curve scalar multiplication on the receiver's side as well as a 160-bit bandwidth overhead in the ciphertext size when compared to a sequential composition of the BLS signature with a CCA2-secure El Gamal encryption.

The present paper aims at proposing a more efficient Diffie-Hellman based signcryption solution that satisfies the same strong security requirements: the scheme that is described here has essentially the same cost as a mere El Gamal encryption on the sender'side while only one pairing evaluation and three exponentiation are required for the simultaneous decryption/verification tasks. This is a real efficiency improvement: the signcryption operation is roughly 33% faster while the de-signcryption algorithm is almost twice more efficient than in Libert and Quisquater's original scheme. The price to pay for such improvements is that our scheme's security relies on stronger assumption than the one depicted in [19]: the chosen-ciphertext security is proved under the q-Diffie-Hellman Inversion assumption already considered in [7], [8] and [35] while the scheme's strong unforgeability relies on the q-Strong Diffie-Hellman assumption introduced by Boneh and Boyen [7].

Before starting with describing our scheme, we have to recall the properties of the bilinear maps that are needed to achieve our purposes. We then discuss about formal security models for signcrytpion and we then give a description of our scheme before discussing about its efficiency and we finally give security proofs in the random oracle model.

2 Preliminaries

2.1 Bilinear maps

Definition 1. *Let k be a security parameter and q be a k-bit prime number. Let us consider groups $(\mathbb{G}_1, +)$, $(\mathbb{G}_2, +)$ and $(\mathbb{G}_T, \cdot)$ of the same prime order p and let P, Q be generators of respectively $\mathbb{G}_1$ and $\mathbb{G}_2$. We say that $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ are*

asymmetric bilinear map groups *if there exists a bilinear map* $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ *satisfying the following properties:*

1. *Bilinearity:* $\forall\ (S,T) \in \mathbb{G}_1 \times \mathbb{G}_2,\ \forall\ a,b \in \mathbb{Z}$, *we have the relation* $e(aS,bT) = e(S,T)^{ab}$.
2. *Non-degeneracy:* $\forall\ S \in \mathbb{G}_1,\ e(S,T) = 1\ \forall\ T \in \mathbb{G}_2$ *iff* $S = 1_{\mathbb{G}_1}$.
3. *Computability:* $\forall\ (S,T) \in \mathbb{G}_1 \times \mathbb{G}_2,\ e(S,T)$ *is efficiently computable.*
4. *There exists an efficient, publicly computable (but not necessarily invertible) isomorphism* $\psi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$ *such that* $\psi(Q) = P$.

Such bilinear map groups are known to be instantiable with ordinary elliptic curves such as MNT curves [24] or a kind of curves studied by Barreto and Naehrig [5].

We now recall some recently considered intractability assumptions that were sometimes called "exponent inversion assumptions".

Definition 2. *Let* $(\mathbb{G}_1, \mathbb{G}_2)$ *be bilinear map groups of generators* P_1 *and* P_2.

- *The* **q-*Diffie-Hellman Inversion problem*** *(q-DHI) consists in, given a tuple* $(P_1, P_2, xP_2, x^2P_2, \ldots, x^qP_2) \in \mathbb{G}_1 \times \mathbb{G}_2^{q+1}$, *computing* $\frac{1}{x}P_1 \in \mathbb{G}_1$.
- *The* **q-*Strong Diffie-Hellman problem*** *(q-SDH) consists in, given a tuple of elements* $(P_1, P_2, xP_2, x^2P_2, \ldots, x^qP_2) \in \mathbb{G}_1 \times \mathbb{G}_2^{q+1}$, *coming up with a pair* $(c, \frac{1}{x+c}P_1) \in \mathbb{Z}_p \times \mathbb{G}_1$.

The q-Diffie-Hellman inversion problem was introduced in [23]. It was also used in [35] where it was shown to be equivalent to the $(q+1)$-exponent problem (namely, finding $x^{q+1}P_1$ given $(P_1, P_2, xP_2, x^2P_2, \ldots, x^qP_2)$). The relaxed q-Strong Diffie Hellman assumption (i.e. the hardness of the q-Strong Diffie-Hellman problem) was introduced in [7] where it was used to prove the security of a new signature scheme in the standard model.

3 Security notions for signcryption schemes

We recall the two usual security notions: the security against chosen ciphertext attacks and the unforgeability against chosen-message attacks. In the former, we consider a multi-user security model as already done in [2, 3, 12, 28, 11] to let the adversary obtain ciphertexts created with the attacked private key under arbitrary public keys. We also consider the "inside attackers" that are allowed to choose the private key under which the challenge is signcrypted: for confidentiality purposes, we require the owner of a private key to be unable to find any information on a ciphertext created with that key without knowing which random coins were 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.

Definition 3. *We say that a signcryption scheme ensures* ***message privacy against chosen-ciphertext attacks*** *(we call this security notion IND-SC-CCA) if no PPT adversary has a non-negligible advantage in the following game:*

1. *The challenger $\mathcal{CH}$ generates 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 of the following kinds:*
   - *Signature/encryption queries: $\mathcal{A}$ produces a message $m \in \mathcal{M}$ and an arbitrary public key pk_R (which may differ from pk_U) and requires the result $\mathsf{Sign/Encrypt}(m, sk_U, pk_R)$ of the signature/encryption oracle.*
   - *Decryption/verification queries: $\mathcal{A}$ produces a ciphertext σ and requires the result of $\mathsf{Decryt/Verify}(\sigma, sk_U)$ which consists of a signed plaintext together with a sender's public key if the obtained signed plaintext is valid for the recovered sender's public key and the $\perp$ symbol otherwise (indicating that the ciphertext was not properly formed).*

   These queries can be asked adaptively: each query may depend on the answers to previous ones. After a number of queries, $\mathcal{A}$ produces two plaintexts m_0, $m_1 \in \mathcal{M}$ and an arbitrary private key sk_S. $\mathcal{CH}$ flips a coin $b \stackrel{R}{\leftarrow} \{0,1\}$ to compute an authenticated encryption $\sigma = \mathsf{Sign/Encrypt}(m_b, sk_S, pk_U)$ of m_b with the sender's private key sk_S under the attacked public key pk_U. The ciphertext σ is sent to $\mathcal{A}$ as a challenge.
3. *$\mathcal{A}$ performs new queries but she may not ask the decryption/verification of the challenge σ. At the end of the game, she outputs a bit b' and wins if $b' = b$.*

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

A lot of provably unforgeable signcryption schemes such as [2, 3, 12, 16, 28] provide non-repudiation with respect to the whole ciphertext. As noticed in [11], in many contexts, it is sufficient to only consider the non-repudiation with respect to the signature embedded in the ciphertext. Even though we still doubt on whether non-repudiation with respect to entire ciphertexts is a relevant or useful security notion, for applications that would be requiring it, we will show to turn our scheme into a ciphertext-existentially unforgeable one at the cost of increasing the size of the detachable signatures by a factor of 3 (but without any other loss of efficiency). The notion of unforgeability w.r.t. embedded signature, that was introduced for the first time in [11], is recalled below.

Definition 4. *We say that a signcryption scheme is said to be **existentially signature-unforgeable** against chosen-message attacks (ESUF-SC-CMA) if no PPT adversary has a non-negligible advantage against a challenger CH in the following game:*

1. *CH generates a key pair (sk_U, pk_U) and pk_U is given to the forger $\mathcal{F}$.*
2. *$\mathcal{F}$ adaptively performs queries to the same oracles as in definition 2.*
3. *$\mathcal{F}$ eventually produces a ciphertext σ and a key pair (sk_R, pk_R) and wins if the result of $\mathsf{Decrypt/Verify}(\sigma, sk_R)$ is a triple (m, s, pk_U) such that the pair (m, s) is valid for the public key pk_U and no query to the signcryption oracle involving the message m and some receiver's public key pk'_R resulted in a ciphertext σ' for which the output of $\mathsf{Decrypt/Verify}(\sigma', sk'_R)$ is (m, s, pk_U).*

As stressed by [11], considering non-repudiation only w.r.t. signatures is useful for schemes providing detachable signatures that should be *unlinkable* to the ciphertext they were conveyed in: anyone seeing a valid message-signature pair can use his/her private key to encrypt it into a valid signcryption under his/her public key. A notion complementary to the latter was called *ciphertext authentication* and means that a receiver is always convinced that a ciphertext was (jointly) signed and encrypted by the same person. We refer to [11] for a formal definition of this notion that is omitted here and that our scheme can be shown to satisfy.

4 The scheme

Common-Keygen given a security parameter k, this algorithm outputs a k-bit prime number p and the description of bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ of order p. Let ℓ_1 and ℓ_2 be polynomials in k respectively denoting the bitlength of elements from $\mathbb{G}_1$ and $\mathbb{G}_2$. The algorithm also chooses generators $P_1 \in \mathbb{G}_1$ and $P_2 \in \mathbb{G}_2$ with $P_1 = \psi(P_2)$, hash functions $h_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p$, $h_2 : \mathbb{G}_1{}^3 \rightarrow \{0,1\}^{k+1}$ and $h_3 : \{0,1\}^k \rightarrow \{0,1\}^\lambda$ as well as a pseudo-random function $h' : \{0,1\}^* \rightarrow \{0,1\}$. The common public parameters are

$$\texttt{param} = \{p, \mathbb{G}_1, \mathbb{G}_2, P_1, P_2, h_1, h_2, h_3, h', n\}$$

where n stands for the length of plaintexts.

Keygen each user picks $x_u \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ and computes $Y_u = x_u P_2 \in \mathbb{G}_2$ obtains a public/private key pair $(pk_u, sk_u) = (Y_u, x_u)$.

Sign/Encrypt given a message $m \in \{0,1\}^n$, the recipient's public key Y_R and her private key x_S, the sender does the following:

1. Pick $\gamma \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ and compute $r = \frac{\gamma}{h_1(b_m||m||Y_S)+x_S} \bmod p$ where $b_m = h'(x_S, m) \in \{0,1\}$.
2. Set $C_1 = rP_1 \in \mathbb{G}_1$, $C_2 = (\gamma||b_m) \oplus h_2\big(C_1||Y_R||r\psi(Y_R)\big) \in \{0,1\}^{k+1}$ and then $C_3 = (m||Y_S) \oplus h_3(\gamma) \in \{0,1\}^{n+\ell_2}$.

The ciphertext is

$$\langle C_1, C_2, C_3 \rangle = \langle rP_1, (\gamma||b_m) \oplus h_2\big(rP_1||Y_R||r\psi(Y_R)\big), (m||Y_S) \oplus h_3(\gamma) \rangle$$

Decrypt/Verify given $C = \langle C_1, C_2, C_3 \rangle$,

1. Compute $(\gamma||b_m) = C_2 \oplus h_2(C_1||Y_R||x_R C_1) \in \{0,1\}^k$.
   Return $\perp$ if $\gamma \notin \mathbb{Z}_p^*$.
2. Compute $(m||Y_S) = C_3 \oplus h_3(\gamma) \in \{0,1\}^{n+\ell_2}$.
3. Compute $\sigma = \gamma^{-1} C_1 \in \mathbb{G}_1$ and accept the message if

$$e\big(\sigma, Y_S + h_1(b_m||m||Y_S)P_2\big) = e(P_1, P_2). \qquad (1)$$

Fig. 1. The q-DH-signcryption scheme

The protocol relies on a signature scheme independently proposed by Zhang et al. [35] and Boneh and Boyen [7]. In the latter papers, this scheme was shown to efficiently produce 160-bit signatures without requiring the use of a special hash function mapping messages to be signed onto an elliptic curve subgroup (unlike the original BLS short signature proposed in [10]). In [7], it was also showed that this scheme has a more efficient security reduction in the random oracle model under the q-strong Diffie-Hellman assumption than the reduction given by Zhang et al. [35] under the q-Diffie-Hellman-Inversion assumption.

The protocol makes use of such a (masked) signature as an ElGamal like ephemeral key as well as a checksum showing that a message was properly encrypted in chosen-ciphertext security concerns: the sender first computes a multiplier $r = \gamma/(h_1(b_m||m||Y_S) + x_S) \in \mathbb{Z}_p^*$ where γ is randomly chosen from $\mathbb{Z}_p^*$, $m \in \{0,1\}^*$ is the message to sign and encrypt and b_m is a message-dependent bit computed as a pseudo-random function of m and the private key x_S according to Katz and Wang's proof technique [18] (that aims at achieving tight security reductions without random salts). This multiplier r is then used to compute an ephemeral Diffie-Hellman key rP_1 as in the ElGamal cryptosystem [13] and to scramble the secret γ using a hash value of $r\psi(Y_R) \in \mathbb{G}_1$ while a digest of γ is used to conceal the message m together with the sender's public key.

The use of a masked signature as a "one-time" Diffie-Hellman key allows sparing one exponentiation (actually an elliptic curve scalar multiplication) w.r.t. a sequential signature/encryption composition.

When computing the second component of the ciphertext, the receiver's public key and the first component (which is an embedded signature as well as a Diffie-Hellman ephemeral key) are hashed together with the "one-time" Diffie-Hellman key $r\psi(Y_R)$ in order to achieve a more efficient reduction in the security proof.

In order to convince a third party that a recovered message m emanates from the sender S, the receiver reveals σ, the message m and the associated bit b_m to the third party who can run the regular signature verification algorithm as done in step 3 of the decryption/verification algorithm. The scheme thus provides detachable signatures that cannot be linked to their original ciphertext: the signature is masked by a randomly chosen factor γ and anyone observing a valid message-signature pair can use his/her private key to build a signcryption of that message-signature pair under his/her public key. Hence, the scheme provides *ciphertext unlinkability* in the sense of Boyen [11] in a very simple manner.

As Boyen's identity based scheme, the present one is obviously not existentially ciphertext-unforgeable in the sense of [19] (because of its inherent ciphertext unlinkability property). We believe that it is actually useless to consider ciphertext non-repudiation (that appears as being antagonist to the useful notion of ciphertext unlinkability and might even be undesirable) rather than the mere signature non-repudiation: an adversary should not be rewarded for achieving the trivial task of using a valid signature and a randomly chosen x'_R as a recipient's private key to output a claimed forged ciphertext under the public key $Y'_R = x'_R P_1$.

Efficiency discussions. As mentioned above, the scheme is efficient since, beside a modular inversion, the sender only computes two scalar multiplications in $\mathbb{G}_1$. The receiver's workload is dominated by one pairing computation (as $e(P_1, P_2)$ can be computed once-and-for-all and cached in memory), two scalar multiplications in $\mathbb{G}_1$ and one multiplication in $\mathbb{G}_2$. For both the sender and the receiver, the protocol is much more efficient than the scheme described in the previous section.

The scheme is described in terms of asymmetric pairings and requires the existence of a publicly computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$. It does not require hashing arbitrary strings onto cyclic elliptic curve subgroups. Hence, the kind of groups suggested in section 4 of [31] may be employed here and allows performing the last step of the $\mathsf{Decrypt/Verify}$ algorithm at a reasonable speed by using specialized techniques for ordinary curves due to Barreto, Lynn and Scott [10].

Interestingly, unlike what appears at first glance, the two exponentiations that are the bulk of the sender's workload can be computed offline (i.e. before knowing the message to be sent). Indeed, in an offline phase, the sender can choose $r \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$, compute $c_1 = rP_1$ and $\omega = r\psi(Y_R)$, store them in memory and then, once the message m is known, compute $\gamma = r(h_1(b_m||m||Y_S) + x_S) \bmod p$, $C_2 = (\gamma||b_m) \oplus h_2(C_1||Y_R||\omega) \in \{0,1\}^k$ and $C_3 = (m||Y_S) \oplus h_3(\gamma)$. In this case, care must be taken not to re-use the same r to sign and encrypt distinct messages because this would expose the private key. This is not a problem since all signatures obtained through the Fiat-Shamir heuristic [14] have this feature. In the absence of reliable pseudo-random generators, the pre-computation phase may choose random powers r as digests (computed using a collision-resistant hash function) of the message and the signer's public/private key pair according to a technique suggested in [25].

Similarly to the Boneh-Boyen signature [7], our scheme can benefit from a great *on-the-fly* efficiency although the signature scheme on which it is constructed does not support the use of "coupons" which is here allowed thanks to the presence of the blinding factor γ .

From a bandwidth point of view, we note that receivers' public keys that are scrambled along with plaintexts in the last part of ciphertexts are $\mathbb{G}_2$ elements of rather long representation. Indeed, for recommended parameters [10] their length is usually at least $\ell = 1024$. However, assuming that the scheme is supported by a public key infrastructure, it is fairly reasonable to encrypt a short (less than 128 ASCII characters) sender-identifier ID_S instead of a public key together with the plaintext in the second step of the $\mathsf{Sign/Encrypt}$ algorithm. The receiver then has to perform an online lookup in a public repository to enquire for the associated public key.

Finally, the present version of the protocol is devised to sign and encrypt messages of bounded size n. In applications that require to handle messages of variable length, the message-public key pair may be symmetrically encrypted using a hash value of γ as a secret key instead of being concealed by a "one-time pad". The employed symmetric encryption scheme then only has to be semantically secure against passive attacks (i.e. attacks must be unable to distinguish two encryptions of plaintexts of their choosing without having access to decryption nor encryption oracles) which is a very weak requirement. The security proof is easily adaptable to the latter hybrid setting.

Short detachable signatures. We have to mention that implementing the scheme with symmetric pairings $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ (and thus supersingular curves) might result in a faster Decrypt/Verify algorithm in some cases. Indeed, the implementation suggested here entails a scalar multiplication in $\mathbb{G}_2$ in step 3 of Decrypt/Verify and arithmetic operations in $\mathbb{G}_2$ are known to be more expensive than in $\mathbb{G}_1$ (recall that, if $\mathbb{G}_1$ is a subgroup of a curve $E(\mathbb{F}_r)$ of embedding degree α, $\mathbb{G}_2$ is usually a cyclic subgroup of $E(\mathbb{F}_{r^\alpha})$ or a subgroup of the twisted curve $\overline{E}(\mathbb{F}_{r^{\alpha/2}})$). For applications where speed is primary concern, using supersingular curves and the group $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$ (where the isomorphism ψ is the identity map) or ordinary curves of low embedding degree (ideally $\alpha = 2$) might be preferable.

On the other hand, ordinary curves of larger embedding degree (say $\alpha = 6$ for instance) should be preferred for applications where bandwidth considerations have priority. Indeed, because of the increased efficiency of particular algorithms for solving discrete logarithms in finite fields of small characteristic, supersingular curves in characteristic 3 are known to offer fewer security than ordinary ones for the same length of group elements. As explained in [10], they are thus not recommended for applications requiring short signatures: examples given in [27] show that, for the security level of a 1024-bit RSA modulus, supersingular curves disallow signatures shorter than 260 bits whereas ordinary MNT curves [24] allow signatures of 171 bits with the Boneh-Lynn-Shacham scheme [10]. For the security level of a 1842-bit RSA modulus, supersingular curves require signatures of 560 bits whereas 308 bits are sufficient with MNT curves.

Those reasons motivated us to describe the scheme in terms of asymmetric pairings. In such an algebraic setting, receivers can extract signatures of 171 bits from ciphertexts. Assuming the deployment of a suitable PKI, ciphertexts may even avoid conveying sender's encrypted public keys: in step 2 of the signature/encryption algorithm, senders may just append their short identifier ID_S to the plaintext. In order to prevent the threat of impersonators replacing ID_S with their own identifier in public repositories, senders may also append ID_S to the plaintext when computing the multiplier r in such a way that C_1 contains a signature on the extended message $m||\mathsf{ID}_S$.

From a computational point of view, all the sender's arithmetic operations take place in $\mathbb{G}_1$ while, besides a pairing evaluation, receivers have to perform a single scalar multiplication in $\mathbb{G}_2$.

5 Security

The original version of the scheme [20] was shown in [33] to be vulnerable to a chosen-ciphertext attack taking advantage of a key substitution attack on the underlying signature scheme [35, 7]. However, protecting the scheme against the attack of [33] was rather straightforward using a standard countermeasure to immunize signature schemes from key substitution attacks: hashing the signer's public key along with the message to be signed suffices.

The security properties called *message confidentiality* and *existential signature unforgeability* respectively rely (in the random oracle model [6]) on the q-DHI and q-SDH assumptions [7, 8] which extend ideas from [23, 29].

Theorem 1. *Assume that an adversary $\mathcal{A}$ has non-negligible advantage ϵ in breaking the IND-SC-CCA security of the scheme when running in time τ, asking q_{h_i} queries to random oracles h_i (for $i = 1, 2, 3$), q_{se} signature/encryption queries and q_{dv} decryption/verification queries. Then there exists a PPT algorithm $\mathcal{B}$ to solve the q-Diffie-Hellman Inversion problem for $q = q_{h_1}$ with advantage*

$$\epsilon' > \epsilon - \frac{q_{dv}}{2^k} - \frac{q_{h_3}}{2^{n+\ell}}$$

when running in time $\tau' < \tau + O(q_{h_1}^2 \tau_m) + 2q_{h_2}\tau_p$ where τ_m is the maximal cost of a scalar multiplication in $\mathbb{G}_1$ and $\mathbb{G}_2$, τ_p being the time for a bilinear map evaluation.

Proof. We actually show how $\mathcal{B}$ can use $\mathcal{A}$ as a subroutine to solve a random instance $(P, Q, xQ, x^2Q, \ldots, x^qQ)$ of the $(q+1)$-exponent problem problem in $(\mathbb{G}_1, \mathbb{G}_2)$ which is equivalent to the q-DHI problem and consists in computing $x^{q+1}P \in \mathbb{G}_1$. We can assume w.l.o.g. that $q_{se} = q_{h_1} - 1$ since, otherwise, $\mathcal{B}$ can issue dummy signcryption queries for itself. In a preparation phase, $\mathcal{B}$ uses its input to compute generators $H \in \mathbb{G}_2$ and $G = \psi(H) \in \mathbb{G}_1$ together with a public key $X = xH \in \mathbb{G}_2$ such that it knows $q_{se} = q - 1$ pairs $(w_i, \frac{1}{w_i + x}G)$ for $w_i \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ as in the technique of [7]. To do so, $\mathcal{B}$ picks $w_1, \ldots, w_{q-1} \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$, expands the polynomial $f(z) = \prod_{i=0}^{q-1}(z + w_i) = \sum_{i=0}^{q-1} c_i z^i$. A generator $H \in \mathbb{G}_2$ and the public key X are then obtained as

$$H = \sum_{i=0}^{q-1} c_i(x^iQ) = f(x)Q \quad \text{and} \quad X = \sum_{i=1}^{q} c_{i-1}(x^iQ) = xf(x)Q = xH$$

(as in the proof of lemma 1 in [7]). As in [7], pairs $(w_i, \frac{1}{w_i+x}G)$ are obtained by expanding $f_i(z) = f(z)/(z + w_i) = \sum_{i=0}^{q-2} d_i z^i$ and computing

$$G_i = \sum_{j=0}^{q-2} d_j \psi(x^jQ) = f_i(x)P = \frac{f(x)}{z + w_i}P = \frac{1}{z + w_i}G$$

for $i = 1, \ldots, q-1$.

The adversary $\mathcal{A}$ is then initialized with the generators $H \in \mathbb{G}_2$ and $G = \psi(H) \in \mathbb{G}_1$ and on the public key $X \in \mathbb{G}_2$. She starts probing the oracles she is given access to. These oracles are simulated by $\mathcal{B}$ as explained below. The queries to oracle h_2 need to be simulated using two lists L_2, L_2' that are initially empty.

- h'-queries on an input $(\alpha_i, m_i) \in \mathbb{Z}_p^* \times \{0,1\}^*$: $\mathcal{B}$ first checks if $X = \alpha_i H$. In this case, we are done and $\mathcal{B}$ can easily compute $x^{q+1}P$. Otherwise, it responds with a random bit $b_{m_i} \xleftarrow{R} \{0,1\}$.
- h_1-queries: these queries are indexed by a counter t that is initially set to 1. When a triple $d||m||X \in \{0,1\} \times \{0,1\}^n \times \mathbb{G}_2$ is submitted in a h_1-query for the first time, $\mathcal{B}$ checks whether d equals the bit b_m (which is set at the first time the message m is submitted in a $h_1(.)$ query). If $d = b_m$, $\mathcal{B}$ returns w_t and increments t (in such a way that $\mathcal{B}$ is able to create a valid signature on m). Otherwise, $\mathcal{B}$ returns a random $c \xleftarrow{R} \mathbb{Z}_p^*$ stores (d, m, X, c) in L_1.
- h_2-queries on input $Y_{1,i}||Y_{2,i}||Y_{3,i} \in \mathbb{G}_1 \times \mathbb{G}_2 \times \mathbb{G}_1$: $\mathcal{B}$ checks if the 4-uple $(H, Y_{1,i}, Y_{2,i}, Y_{3,i})$ is a valid co-Diffie-Hellman tuple (in our notation, we shall write $Y_{3,i} = \text{co-DH}_H(Y_{1,i}, Y_{2,i})$) by verifying if

$$e(Y_{1,i}, Y_{2,i}) = e(Y_{3,i}, H).$$

  If it is, $\mathcal{B}$ checks if L_2' contains a record $(Y_{1,i}, Y_{2,i}, ., \tau_i)$ for some $\tau_i \in \{0,1\}^{k+1}$. In this case, τ_i is returned and an entry $(Y_{1,i}, Y_{2,i}, Y_{3,i}, \tau_i, 1)$ is added in L_2. If no entry of the shape $(Y_{1,i}, Y_{2,i}, ., \tau_i)$ is in L_2', $\mathcal{B}$ returns a string $\tau_i \xleftarrow{R} \{0,1\}^{k+1}$ and inserts $(Y_{1,i}, Y_{2,i}, Y_{3,i}, \tau_i, 1)$ in L_2. If $(H, Y_{1,i}, Y_{2,i}, Y_{3,i})$ is not a co-DH tuple, the entry $(Y_{1,i}, Y_{2,i}, Y_{3,i}, \tau_i, 0)$ is added in L_2. At most $2q_{h_2}$ pairings must be computed overall.
- h_3-queries: are answered with random elements from $\{0,1\}^{n+\ell_2}$.
- Signature/encryption queries on a plaintext m, for an arbitrary receiver's key Y: we assume that m was previously submitted in a h_1-query and that the message-dependent bit b_m was previously defined. Since $h_1(b_m||m||X)$ was (or will be) defined to be w_j for some $j \in \{1, \dots, t\}$, $\mathcal{B}$ knows that the previously computed $G_j = (1/(w_j + x))G$ appears as a valid signature on m from $\mathcal{A}$'s view. So, it computes $C_1 = \gamma G_j \in \mathbb{G}_1$ for some $\gamma \xleftarrow{R} \mathbb{Z}_p^*$, obtains $\kappa = h_3(\gamma) \in \{0,1\}^{n+\ell_2}$ through h_3-simulation and computes $C_3 = (m||X) \oplus \kappa \in \{0,1\}^{n+\ell_2}$. It then checks if L_2 contains an entry $(C_1, Y, Y_3, \tau, 1)$ (indicating that $Y_3 = \text{co-DH}_H(C_1, Y)$). If this entry exists, $\mathcal{B}$ returns $C = \langle C_1, C_2, C_3 \rangle$ with $C_2 = (\gamma||b_m) \oplus \tau \in \{0,1\}^{k+1}$. Otherwise it returns $C = \langle C_1, C_2, C_3 \rangle$ for a random $C_2 \xleftarrow{R} \{0,1\}^{k+1}$ and inserts $\big(C_1, Y, ., (\gamma||b_m) \oplus C_2\big)$ in the special list L_2'.
- Decryption/verification queries: when $\mathcal{A}$ submits a ciphertext $C = \langle C_1, C_2, C_3 \rangle$, $\mathcal{B}$ checks whether L_2 contains the unique entry $(C_1, X, Y, \tau, 1)$ for some $Y \in \mathbb{G}_1$ and $\tau \in \{0,1\}^{k+1}$ (indicating that $Y = \text{co-DH}_H(C_1, X)$):
  - if it does, $\mathcal{B}$ obtains $(\gamma||b_m) = C_2 \oplus \tau \in \{0,1\}^{k+1}$, $\kappa = h_3(\gamma)$ (via simulation of h_3) and finally $(m||X_S) = C_3 \oplus \kappa \in \{0,1\}^{n+\ell_2}$ (C is also rejected if X_S is not a $\mathbb{G}_2$ element). Finally, $\mathcal{B}$ extracts $\sigma = \gamma^{-1}C_1$ and returns the plaintext $m \in \{0,1\}^n$ and the signature σ together with the sender's public key $X_S \in \mathbb{G}_2$ if the verification equation (1) holds.
  - if it does not, $\mathcal{B}$ picks a random $\tau \xleftarrow{R} \{0,1\}^{k+1}$, inserts $(C_1, X, ., \tau)$ into the list L_2' (so that a subsequent h_2-query on $(C_1, X, \text{co-DH}_H(C_1, X))$ will receive τ as an answer), before finishing the job with the randomly

chosen τ: it computes $(\gamma||b_m) = C_2 \oplus \tau \in \{0,1\}^{k+1}$, $\kappa = h_3(\gamma)$, and so on. The extracted signature $\sigma = \gamma^{-1}C_1$ is checked as above.

After the find stage, $\mathcal{A}$ outputs messages m_0, m_1 and a sender's private key $x_S \in \mathbb{Z}_p^*$. At this moment, $\mathcal{B}$ chooses a random $a \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ and computes $C_1^* = (x+a)G \in \mathbb{G}_1$ as $C_1^* = \psi(X) + aG$. It also expands the polynomial $f'(z) = f(z)(z+a) = \sum_{j=0}^{q} f_j z^j \in \mathbb{Z}_p[z]$ and returns the challenge $C^* = \langle C_1^*, C_2^*, C_3^* \rangle$, where $b^* \stackrel{R}{\leftarrow} \{0,1\}$, $C_2^* \stackrel{R}{\leftarrow} \{0,1\}^{k+1}$ and $C_3^* = (m_d||x_S H) \oplus \kappa$ for a random bit $d \stackrel{R}{\leftarrow} \{0,1\}$ and a random $\kappa \stackrel{R}{\leftarrow} \{0,1\}^{n+\ell_2}$. In the extremely unlikely case (its probability is much smaller than $q_{dv}/2^k$) that C^* was submitted in a $\mathsf{Decrypt/Verify}$ query before the challenge phase, $\mathcal{B}$ aborts.

Clearly, if κ does not hit the output of an h_3-query (the probability for this to occur is at most $q_{h_3}/2^{n+\ell_2}$), the adversary's view is independent from the bit d unless the hash value $h_2\big(C_1^*||X||\text{co-DH}_H(C_1^*, X)\big)$ is asked during the simulation. Such an event, that we call AskH_2, is easily detected by the h_2 simulator and is very likely to happen: one can easily show that in a real attack, $\Pr[\mathsf{AskH}_2]$ is at least ϵ if $\mathcal{A}$'s advantage in definition 3 is ϵ. Besides, as long as $\mathcal{A}$ is provided with a consistent view, $\Pr[\mathsf{AskH}_2]$ is the same in the simulation as in the real world.

Queries made by $\mathcal{A}$ in the second stage are handled as above and, as already argued, the h_2-simulator must detect the AskH_2 event with high probability. At this moment, $\mathcal{B}$ obtains $Z = \text{co-DH}_H(C_1^*, X) = x(x+a)G = xf(x)(x+a)P$. Since $f(z)(z+a)z = zf'(z) = \sum_{j=0}^{q} f_j z^{j+1}$ and, since $Z = \sum_{j=0}^{q} f_j \psi(x^{j+1}Q)$, $\mathcal{B}$ can compute

$$(x^{q+1})P = \frac{1}{f_q}\big[Z - \sum_{j=0}^{q-1} f_j \psi(x^{j+1}Q)\big] \in \mathbb{G}_1$$

which solves the $(q+1)$-exponent instance. At this stage, we are done since the latter is known to be equivalent to the q-Diffie-Hellman Inversion problem (as explained in [8]).

From a computational point of view, $\mathcal{B}$'s running time is dominated by $q+2$ multi-exponentiations with q elements that reach an overall cost of $O(q^2)$ exponentiations. Computing $f(z)$ also involves a cost in $O(q^2)$ while computing each $f_i(z)$ also implies $O(q)$ modular multiplications just like the computation of the product $f(z)(z+a)$. When handling h_2 queries, $\mathcal{B}$ also has to compute $2q_{h_2}$ pairings.

The bound on $\mathcal{B}$'s advantage derives from the fact that it never provides $\mathcal{A}$ with an incoherent view. The simulation only fails at the challenge phase if the selected κ hits the output of a previous h_3-query or if the constructed challenge was previously submitted to the $\mathsf{Decrypt/Verify}$ oracle. $\square$

We observe that the reduction is tight in terms of probabilities in that the probability of solving the computational problem is close to the adversary's advantage. Unfortunately, the reduction is not particularly efficient as its cost is bounded by an expression containing a term which is quadratic in the number of adversarial random oracle queries. With the bound $q_{h_1} < 2^{60}$ usually considered in the literature, our reduction does not lead to a strong concrete security.

Nevertheless, we do not see this lack of efficiency in the reduction as a serious problem. After all, a similar remark can be made on the security reduction given in [7] for the underlying signature scheme in the random oracle model.

Theorem 2. *If an ESUF-SC-CMA adversary $\mathcal{F}$ has a non-negligible advantage ϵ in the game of definition 4 when running in time τ, making q_{h_i} queries to oracles h_i $(i = 1, 2, 3)$, q_{dv} Decrypt/Verify queries and q_{se} Sign/Encrypt queries, then there exists an algorithm $\mathcal{B}$ that solves the q-strong Diffie-Hellman problem for $q = q_{h_1}$ with probability*

$$\epsilon' > \frac{\epsilon}{2} - \frac{1}{2^k} - \frac{1}{2^{n+\ell}}$$

within time $\tau' < \tau + O(q_{h_1}^2 \tau_m) + 2q_{h_2}\tau_p$ where τ_m is the maximal cost of a scalar multiplication in $\mathbb{G}_1$ and $\mathbb{G}_2$, τ_p being the time for a bilinear map evaluation.

Proof. We build a simulator $\mathcal{B}$ that behaves almost exactly as in the previous proof. The generator $G \in \mathbb{G}_1$, that is given to the forger $\mathcal{F}$ as a part of the output of the common key generation algorithm, is generated as in the proof of theorem 1 so that the simulator $\mathcal{B}$ knows $q-1$ pairs $(w_i, \frac{1}{w_i+x}G)$ (where $x \in \mathbb{Z}_p^*$ is the unknown element that implicitly defines its input $P, Q, xQ, x^2Q, \ldots, x^qQ$). By doing so, $\mathcal{B}$ is always able to answer signature/encryption queries that are handled, as all other oracle queries, exactly as in the proof of theorem 1.

Eventually, $\mathcal{F}$ halts and outputs a forged signature embedded into a ciphertext $C^* = \langle C_1^*, C_2^*, C_3^* \rangle$ and an arbitrary recipient's key pair $(x_R^*, Y_R^* = x_R^* H)$ that allows $\mathcal{B}$ recovering the fake pair $(m^*, \sigma^* = \frac{1}{h_1(b^*||m^*||X)+x}G)$ embedded into C^*. With a probability 1/2, b^* differs from the message-dependent bit b_{m^*} (that indicates how a message the message m^* should be signed with the private key corresponding to X in the underlying signature scheme and that is independent from $\mathcal{F}$'s view) and $\mathcal{B}$ can extract a solution to the q-SDH problem as follows: if $\mathcal{F}$ is successful, $\mathcal{B}$ recovers a valid message-signature pair for the sender's public key X by computing $(\gamma^*||b^*) = C_2 \oplus h_2(C_1^*||Y_R^*||x_R^* C_1^*)$, $(m^*||X) = C_3^* \oplus h_3(\gamma^*)$ and $\sigma^* = {\gamma^*}^{-1} C_1^*$. A q-SDH pair $\langle h_1^*, G^* \rangle$ can then be extracted by expanding $f(z)/(z + h_1^*)$ into

$$\frac{\gamma_{-1}}{z + h_1^*} + \sum_{i=0}^{q-2} \gamma_i z^i,$$

where $h_1^* = h_1(b^*, m^*)$, and computing $G^* = \frac{1}{\gamma_{-1}} \big[\sigma^* - \sum_{i=0}^{q-2} \gamma_i \psi(x^i Q)\big]$. A lower bound on the simulator's probability to obtain it is thus one half of the advantage of the simulator of the previous proof decreased by the (negligible) probability for $\mathcal{F}$ to produce a valid encryption of the fake signature without asking the appropriate h_2 and h_3 values during the simulation (in the latter case, $\mathcal{B}$ is unable to extract a q-SDH pair). □

The ciphertext anonymity property can be proved very easily under the q-DHI assumption. We were unfortunately unable to formally establish that the scheme satisfies the ciphertext authentication property in the sense of Boyen's definition [11]. The reason is that, using the proof technique of theorems 1 and 2, it is difficult to set out a generator and two distinct public keys for which a series of signature/encryption queries must be simulated for both public keys without knowing the matching private keys.

Nevertheless, the scheme does seem to meet that property. Given a message-signature pair, it seems to be infeasible for anyone but the signer to encrypt it into a ciphertext intended for a specific receiver without knowing the latter's private key.

Adapting the scheme to applications that would require the ciphertext unforgeability is straightforward: the first part of the ciphertext must be $C_1 = (1/(h_1(b_m||m||\gamma||Y_S||Y_R) + x_S))P$ where $\gamma \in \mathbb{Z}_p^*$ is encrypted in the component C_2 and Y_R is the receiver's public key.

As a consequence, γ and Y_R become part of the detached signature and the sender of a message is not only committed to the plaintext's content but he/she is also responsible for having sent it to the owner of the key Y_R.

6 Conclusion

We presented an efficient signcryption scheme based on discrete logarithm related assumptions and we proved its security in the random oracle model. The scheme was shown to have a great online efficiency (similarly to the basic El Gamal cryptosystem but unlike a Fujisaki-Okamoto/El Gamal encryption scheme).

As an observation of independent interest, we extended our method to achieve this online efficiency to the Boneh-Boyen short signature [7].

References

1. J.-H. An, *Authenticated Encryption in the Public-Key Setting: Security Notions and Analyses*, eprint available at http://eprint.iacr.org/2001/079/, 2001.
2. J.-H. An, Y. Dodis, T. Rabin, *On the security of joint signature and encryption*, Advances in Cryptology - Eurocrypt'02, LNCS 2332, pp. 83–107, Springer, 2002.
3. J. Baek, R. Steinfeld, Y. Zheng, *Formal Proofs for the Security of Signcryption*, PKC'02, LNCS 2274, pp. 80–98, Springer, 2002.
4. F. Bao, R.-H. Deng, *A signcryption scheme with signature directly verifiable by public key*, PKC'98, LNCS 1988, pp. 55–59, 1998.
5. P. S. L. M. Barreto and M. Naehrig. Pairing-friendly elliptic curves of prime order. In *SAC'05*. To Appear.
6. M. Bellare, P. Rogaway, *Random oracles are practical: A paradigm for designing efficient protocols*, 1^{st} ACM Conference on Computer and Communications Security, pp. 62-73, 1993.
7. D. Boneh, X. Boyen, *Short Signatures Without Random Oracles*, Advances in Cryptology - Eurocrypt'04, LNCS 3027, Springer, pp. 56–73, 2004.
8. D. Boneh, X. Boyen, *Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles*, Advances in Cryptology - Eurocrypt'04, LNCS 3027, Springer,pp. 223–238, 2004.

9. D. Boneh, M. Franklin, *Identity Based Encryption From the Weil Pairing*, Advances in Cryptology - Crypto'01, LNCS 2139, pp. 213–229, Springer, 2001.
10. D. Boneh, B. Lynn, H. Shacham, *Short signatures from the Weil pairing*, Advances in Cryptology - Asiacrypt'01, LNCS 2248, pp. 514–532, Springer, 2001.
11. X. Boyen, *Multipurpose identity-based signcryption: A swiss army knife for identity-based cryptography*, Advances in Cryptology (CRYPTO '03), LNCS 2729, pp. 382–398, Springer, 2003.
12. Y. Dodis, M.-J. Freedman, S. Jarecki, S. Walfish, *Versatile Padding Schemes for Joint Signature and Encryption*, to appear at ACM Conference on Computer and Communication Security (CCS), October 2004.
13. T. El Gamal, *A Public Key Cryptosystem and Signature Scheme Based on Discrete Logarithms*, IEEE Trans. on Information Theory, vol. 31, 1985.
14. A. Fiat, A. Shamir, *How to Prove Yourself: Practical Solutions to Identification and Signature Problems* , Advances in Cryptology - Crypto'86, LNCS 0263, Springer, pp. 186-194, 1986.
15. E. Fujisaki, T. Okamoto, *Secure integration of asymmetric and symmetric encryption schemes*, Advances in Cryptology - Crypto'99, LNCS 1666, pp. 537–554, Springer, 1999.
16. I.-R. Jeong, H.-Y. Jeong, H.-S. Rhee, D.-H. Lee, I.-L. Jong, *Provably secure encrypt-then-sign composition in hybrid signcryption*, ICISC'02, LNCS 2587, 2002.
17. A. Joux, K. Nguyen. *Separating Decision Diffie-Hellman from Diffie-Hellman in cryptographic groups*, available at http://eprint.iacr.org/2001/003/, 2001.
18. J. Katz, N. Wang, *Efficiency improvements for signature schemes with tight security reductions*, 10^{th} ACM Conference on Computer and Communications Security, pp. 155–164, 2003.
19. B. Libert, J.-J. Quisquater, *Efficient Signcryption with Key Privacy from Gap Diffie-Hellman Groups*, PKC'04, LNCS 2947, pp. 187-200, Springer, 2004.
20. B. Libert and J. J. Quisquater. Improved signcryption from q-Diffie-Hellman problems. In *SCN'04*, volume 3352 of *LNCS*, pages 220–234. Springer, 2004.
21. J. Malone-Lee, *Signcryption with non-repudiation*, Technical report available at http://www.cs.bris.ac.uk/Tools/Reports/Ps/2002-malonelee.pdf, 2002.
22. J. Malone-Lee, W. Mao, *Two birds one stone: Signcryption using RSA*, Topics in Cryptology - CT-RSA 2003, LNCS 2612, pp. 211-225, Springer, 2003.
23. S. Mitsunari, R. Sakai, M. Kasahara, *A new traitor tracing*, IEICE Trans. Vol. E85-A, No.2, pp. 481-484, 2002.
24. A. Miyaji, M. Nakabayashi, S. Tanako, *New Explicit Conditions of Elliptic Curve Traces for FR-Reduction.* IEICE Trans. Fundamentals, E84-A(5):1234-43, 2001.
25. D. M'Raïhi, D. Naccache, D. Pointcheval, S. Vaudenay, *Computational Alternatives to Random Number Generators*, Selected Areas in Cryptography'98 (SAC'98), LNCS 1566, pp. 72-80, Springer, 1998.
26. T. Okamoto, D. Pointcheval, *The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes*, PKC'01, LNCS 1992, Springer, 2001.
27. D. Page, N. P. Smart, and F. Vercauteren. A comparison of MNT curves and supersingular curves. Cryptology ePrint Archive, Report 2004/165, 2004. `http://eprint.iacr.org/2004/165`.
28. J. Pieprzyk, D. Pointcheval, *Parallel Authentication and Public-Key Encryption*, ACISP'03, LNCS 2727, pp. 383–401, Springer, 2003.
29. R. Sakai and M. Kasahara. ID-based cryptosystems with pairing on elliptic curve. In *SCIS'03*, Hamamatsu, Japan, 2003. `http://eprint.iacr.org/2003/054`.
30. J.-B. Shin, K. Lee, K. Shim, *New DSA-verifiable signcryption schemes*, ICISC'02, LNCS 2587, pp. 35–47, Springer, 2002.

31. N. P. Smart and F. Vercauteren. On computable isomorphisms in efficient pairing based systems. Cryptology ePrint Archive, Report 2005/116, 2005. `http://eprint.iacr.org/2005/116`.
32. R. Steinfeld, Y. Zheng, *A signcryption scheme based on integer factorization*, ISW'00, LNCS 1975, pp. 308–322, Springer, 2000.
33. C. H. Tan. Security analysis of signcryption scheme from q-DiffieHellman problems. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E89-A(1):206208, 2006.
34. D.-H. Yum, P.-J. Lee, *New signcryption schemes based on KCDSA*, ICISC'01, LNCS 2288, pp. 305–317, Springer, 2001.
35. F. Zhang, R. Safavi-Naini, W. Susilo, *An Efficient Signature Scheme from Bilinear Pairings and Its Applications*, PKC'04, LNCS 2947, pp. 277–290, 2004.
36. Y. Zheng, *Digital signcryption or how to achieve cost (signature & encryption) << cost(signature) + cost(encryption)*, Advances in Cryptology - Crypto'97, LNCS 1294, pp. 165–179, 1997.