On Generic Constructions of Designated Confirmer Signatures

(The "Encryption of a Signature" Paradigm Revisited)

Anonymous submission to IndoCrypt'09

No Institute Given

Abstract. Designated Confirmer signatures were introduced to limit the verification property inherent to digital signatures. In fact, the verification in these signatures is replaced by a confirmation/denial protocol between the *designated confirmer* and some verifier. An intuitive way to obtain such signatures consists in first generating a digital signature on the message to be signed, then encrypting the result using a suitable encryption scheme. This approach, referred to as the "encryption of a signature" paradigm, requires the constituents (encryption and signature schemes) to meet the highest security notions in order to achieve secure constructions.
In this paper, we revisit this method and establish the necessary and sufficient assumptions on the building blocks in order to attain secure confirmer signatures. Our study concludes that the paradigm, used in its basic form, cannot allow a class of encryption schemes, which is vital for the efficiency of the confirmation/denial protocol. Next, we consider a slight variation of the paradigm, proposed in the context of undeniable signatures; we recast it in the confirmer signature framework along with changes that yield more flexibility, and we demonstrate its efficiency by explicitly describing its confirmation/denial protocol when instantiated with building blocks from a large class of signature/encryption schemes. Interestingly, the class of signatures we consider is very popular and has been for instance used to build efficient designated verifier signatures.

Keywords: Designated Confirmer signatures, "Encryption of a signature" paradigm, Generic construction, Reduction/meta-reduction, Zero Knowledge.

1 Introduction

Digital signatures capture most of the properties met by signatures in the paper world, for instance, universal verification. However, in some applications, this property is not desired or at least needs to be controlled. Undeniable signatures were introduced [12] for this purpose; they proved critical in situations where privacy or anonymity is a big concern, such as licensing software [12], electronic cash and electronic voting and auctions. In these signatures, the verification can be only attained by means of a cooperation with the signer, called the confirmation/denial protocol. Unfortunately, this very virtue (verification with only the signer's help) became its major shortcoming for many practical applications. The flaw was later repaired in [10] by introducing the concept of *designated confirmer signatures*. In fact, this concept involves three entities, namely the signer who produces the signature, the designated confirmer who confirms or denies an alleged signature and finally the recipient of the signature. Designated confirmer signatures, or confirmer signatures for brevity, can have the additional feature of being converted, by the confirmer, to ordinary digital signatures.

1.1 Related work

Since the introduction of confirmer signatures, researchers sought ways of producing them from digital signatures and other cryptographic primitives such as encryption and/or commitment schemes. We briefly review in this paragraph, in chronological order, the most important such attempts:

Okamoto (1994) [32]. The result proposes a construction of confirmer signatures from digital signatures, public key encryption, bit-commitment schemes and pseudo-random functions. The construction was used to prove equivalence between confirmer signatures and public key encryption with respect to existence. Thus, efficiency was not taken into account in the framework.

Michels and Stadler (1998) [30]. This approach builds efficient confirmer signatures from signatures obtained from the Fiat-Shamir paradigm and from commitment schemes. Thus, The resulting confirmer signatures can be only proven secure in the random oracle model (ROM), inheriting this property from the use of the Fiat-Shamir paradigm, which constitutes their major shortcoming. Actually, it is well known, according to [38], that most discrete-logarithm-based signatures obtained from the Fiat-Shamir technique are very unlikely to preserve the same level of security in the standard model.

Camenisch and Michels (2000) [8]. The authors present the "encryption of signature" idea along with a security analysis of the resulting confirmer signatures. In fact, they require existentially unforgeable signatures and indistinguishable encryption in the strongest attack model (EUF-CMA signatures and IND-CCA secure encryption) to achieve unforgeable, invisible, and transcript-simulatable confirmer signatures. The major weakness of the construction lies in the resort, in the confirmation/denial protocol, to general concurrent zero knowledge (ZK) protocols of NP statements.

Goldwasser and Waisbard (2004) [23]. This result manages to circumvent partially the weakness of the above construction. In fact, from a large class of digital signatures, the authors propose a transformation to confirmer signatures by encrypting the former items under an IND-CCA secure encryption during the confirmation protocol. They consequently achieve an efficient confirmation, but at the expense of the transcript-simulatability, the invisibility and the length of the resulting signatures. For instance, the signature contains at least twice the number of the confirmation protocol's rounds of encryptions. Moreover, the denial protocol of the construction has still recourse to general concurrent ZK protocols of NP statements.

Gentry et al. (2005) [19]. This work gives the possibility of building confirmer signatures from digital signatures, encryption (IND-CCA) and commitment schemes. Although the resulting construction does not use random oracles, it still does not get rid completely of general ZK proofs since the confirmer has to prove in concurrent ZK the knowledge of the decryption of an IND-CCA encryption and of a string used for commitment.

Wang et al. (2007) [44]. In this work, the authors present two constructions. The first one fixes some flaws noticed in [19], however, it still requires concurrent ZK the knowledge of NP statements. The second construction does not require any encryption, but at the expense of the underlying security assumption. In fact, it has its invisibility resting on the decisional Diffie-Hellman assumption, which rules out using the scheme in bilinear groups and thus benefiting from the attractive features they present such as achieving short group elements. Moreover, the construction suffers also the recourse to the ROM. Finally, these constructions as well as the construction in [19] are not anonymous, as we will point later in this document.

El Aimani (2008) [14]. This construction is a slight variation of the "encryption of a signature" paradigm which uses cryptosystems from the KEM/DEM paradigm and requires them to be only IND-CPA secure. The author claims that this impacts positively the efficiency of the confirmation/denial protocols by allowing homomorphic schemes in the design. However, such a claim lacks justification since the only illustrations provided in the paper (or in its full version [28]) are generic constructions from a class of pairing-based signatures, which are used with a specific cryptosystem (El Gamal encryption or the linear Diffie-Hellman KEM-DEM). Furthermore, one of the constructions uses a cryptosystem which operates on messages from $\mathbb{Z}_p^\times$ (for some prime p), thus, the resulting signatures will be quite long because of the size contrast between ring cryptography and elliptic-curve cryptography. This seems to violate the main expectation from appealing to elliptic curve cryptography, namely achieve short signatures.

Summing up the state-of-the art in confirmer signatures, we deduce that the most mountainous obstacle that faces the potentially anonymous generic constructions without ROM, namely those derived from variants of the "encryption of a signature" paradigm, lies in the resort to general zero knowledge (ZK) proofs of NP statements, e.g., proving in ZK the knowledge of the decryption of an IND-CCA encryption. In this paper, we revisit this paradigm. We basically address two questions: does the paradigm, used in its basic form [8], allow building blocks with weaker security assumptions, for instance IND-CPA cryptosystems and thus achieves efficient signatures as claimed in [14]? The second question concerns the alleged efficiency of the construction in [14]; how important is the contribution of the IND-CPA requirement to the efficiency of the confirmation/denial protocol?

1.2 Our contributions

The results in this paper are twofold. First, we consider the plain "encryption of a signature" paradigm as described in [8]. We actually prove that EUF-CMA secure signatures are a sufficient and necessary requirement to obtain EUF-CMA secure convertible confirmer signatures. Next, we show that indistinguishable cryptosystems under a *plaintext checking attack* (IND-PCA) are already enough to obtain invisible signatures under a chosen message attack (INV-CMA). This contrasts the wide belief that the cryptosystems should be IND-CCA secure. We also show that this assumption on the cryptosystem (IND-PCA secure) is necessary to obtain invisible signatures. This rules out automatically from the design homomorphic cryptosystems, a class of cryptosystems which proved later to be vital for the efficiency of the confirmation/denial protocol.

Next, we consider the proposal in [14] which builds a universally convertible undeniable signature scheme from secure digital signatures and IND-CPA secure cryptosystems obtained from the KEM/DEM paradigm. We propose a recast of the construction in the confirmer signature framework and we demonstrate its efficiency by explicitly describing the confirmation/denial protocol when instantiated with a large class of signature/encryption schemes. Interestingly, the class of signatures we consider has been already defined as an ingredient of an efficient construction of designated verifier signatures [41]. We conclude that our recast of [14] betters the previous constructions of confirmer signatures in terms of both efficiency and security. In fact, it gets rid of general ZK protocols of NP statements in the confirmation and/or the denial protocol, oppositely to the constructions in [32, 8, 23, 19, 44]. Moreover, the resulting signatures are not proven secure in the random oracle as in [30, 44], and they enjoy a strong invisibility which captures both the traditional invisibility, defined [8], and anonymity which was defined later in [17]. We prove for instance that the latter property is not met by the constructions in [19, 44].

2 Convertible Designated Confirmer Signatures (CDCS)

Since their introduction, many definitions and security models for CDCS have emerged. We consider the default model adopted in most confirmer signature proposals [8, 23, 19, 44, 14]. This model was primally described in [8], where the sign_then_encrypt technique was first formally introduced.

We refer to Appendix A for the necessary cryptographic primitives that will come into use, that are, digital signatures, public key encryption schemes, KEM/DEM mechanisms, and finally Σ protocols.

2.1 Syntax

A CDCS scheme consists of the following procedures:

Key generation. Generates probabilistically key pairs $(\mathsf{sk}_S, \mathsf{pk}_S)$ and $(\mathsf{sk}_C, \mathsf{pk}_C)$ for the signer and for the confirmer respectively, consisting of the private and the public key.

ConfirmSign. On input sk_S, pk_C and a message m, outputs a confirmer signature signature μ, then interacts with the signature recipient to convince him of the validity of the just generated signature.

Confirmation/Denial protocol. These are interactive protocols between the confirmer and a verifier. Their common input consists of, in addition to pk_S and pk_C, the alleged signature μ, and the message m in question. The confirmer uses his private key sk_C to convince the verifier of the validity (invalidity) of the signature μ on m. At the end, the verifier either accepts or rejects the proof.

Selective conversion. This is an algorithm run by the confirmer using sk_C, in addition to pk_C and pk_S. The result is either $\perp$ or a string which can be universally verified as a valid digital signature.

Selective verification. This is an algorithm for verifying converted signatures. It inputs the converted signature, the message and pk_S and outputs either 0 or 1.

2.2 Security model.

The above algorithms must be complete. Moreover the confirmSign, confirmation and denial protocols must be complete, sound and non transferable (simulatable) (see [8]). In the sequel, we describe further properties that a CDCS scheme should meet.

Security for the signer (unforgeability). It is defined through the following game: the adversary $\mathcal{A}$ is given the public parameters of the CDCS scheme, namely pk_S and pk_C, in addition to sk_C. $\mathcal{A}$ is further allowed to query the signer on polynomially many messages, say q_s. At the end, $\mathcal{A}$ outputs a pair consisting of a message m, that has not been queried yet, and a string μ. $\mathcal{A}$ wins the game if μ is a valid confirmer signature on m. We say that a CDCS scheme is (t, ϵ, q_s)-EUF-CMA secure if there is no adversary, operating in time t, that wins the above game with probability greater than ϵ.

Security for the confirmer (invisibility). Invisibility against a chosen message attack (INV1-CMA) is defined through the following game between an attacker $\mathcal{A}$ and his challenger $\mathcal{R}$: after $\mathcal{A}$ gets the public parameters of the scheme from $\mathcal{R}$, he starts **Phase 1** where he queries the signing, confirmation/denial, selective conversion oracles in an adaptive way. Once $\mathcal{A}$ decides that **Phase 1** is over, he outputs two messages m_0, m_1 that have not been queried before to the signing oracle and requests a challenge signature $\mu^\star$. $\mathcal{R}$ picks uniformly at random a bit $b \in \{0,1\}$. Then $\mu^\star$ is generated using the signing oracle on the message m_b. Next, $\mathcal{A}$ starts adaptively querying the previous oracles (**Phase 2**), with the exception of not querying m_0, m_1 to the signing oracle and $(m_i, \mu^\star)$, $i = 0, 1$, to the confirmation/denial and selective conversion oracles. At the end, $\mathcal{A}$ outputs a bit b'. He wins the game if $b = b'$. We define $\mathcal{A}$'s advantage as $\mathsf{adv}(\mathcal{A}) = |\Pr[b = b'] - \frac{1}{2}|$. We say that a CDCS scheme is $(t, \epsilon, q_s, q_v, q_{sc})$-INV1-CMA secure if no adversary operating in time t, issuing q_s queries to the signing oracle, q_v queries to the confirmation/denial oracles and q_{sc} queries to the selective conversion oracle wins the above game with advantage greater that ϵ.

Anonymity of signatures. In some applications, it is required that the confirmer signatures are anonymous, i.e., do not leak the identity (public key) of the signer. We refer to [17] for the formal definition of anonymity of confirmer signature under a chosen message attack (ANO-CMA).

A stronger notion of invisibility. To capture both anonymity and invisibility, Galbraith and Mao introduced in [17] a notion, which we denote INV2-CMA, that requires the confirmer signatures to be indistinguishable from random elements in the signature space. This new notion is proven to imply both INV1-CMA and ANO-CMA (Theorem 1 and Theorem 4 respectively of [17]).

3 The Plain "Encryption of a Signature" Paradigm

The paradigm devises a CDCS scheme by producing a digital signature on the message to be signed, then encrypting the result using a suitable cryptosystem. More precisely, Let Σ be a digital signature scheme given by $\Sigma.\mathsf{keygen}$ which generates a key pair (private key = $\Sigma.\mathsf{sk}$, public key= $\Sigma.\mathsf{pk}$), $\Sigma.\mathsf{sign}$ and $\Sigma.\mathsf{verify}$. Let furthermore Γ denote a cryptosystem described by $\Gamma.\mathsf{keygen}$ that generates the key pair (private key = $\Gamma.\mathsf{sk}$, public key= $\Gamma.\mathsf{pk}$), $\Gamma.\mathsf{encrypt}$ and $\Gamma.\mathsf{decrypt}$. A confirmer signature on a message m is issued by first producing a digital signature $\sigma = \Sigma.\mathsf{sign}_{\Sigma.\mathsf{sk}}(m)$ on m, then encrypting it using $\Gamma.\mathsf{pk}$. The result is $\mu = \Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(\sigma)$. It is obvious that $\Sigma.\mathsf{sk}$ forms the (DCSC) signer's private key, whereas $\Sigma.\mathsf{pk}$ is his public key. To confirm (deny) a confirmer signature μ, the confirmer uses $\Gamma.\mathsf{sk}$ to prove the knowledge of the decryption of μ which does (not) satisfy the equation defined by the algorithm $\Sigma.\mathsf{verify}$. Such a proof of knowledge is possible as the considered statements are in NP (co-NP), and therefore accept zero knowledge proof systems (see [20]).

This technique was described and formally analyzed in [8]: it was shown that the construction is EUF-CMA secure if the underlying (digital) signature scheme is also EUF-CMA secure. Moreover, it is INV1-CMA secure if the underlying cryptosystem is IND-CCA secure. Finally, completeness, soundness and non-transferability of the involved protocols follow from using ZK proofs of knowledge.

In the sequel, we prove that the condition on the underlying signature scheme (EUF-CMA secure) is also necessary to achieve EUF-CMA secure confirmer signatures. Furthermore, we prove that IND-PCA secure cryptosystems are already enough, though mandatory, to achieve INV1-CMA signatures.

Theorem 1. *The above generic construction is (t, ϵ, q_s)-EUF-CMA secure if and only if the underlying digital signature scheme is (t, ϵ, q_s)-EUF-CMA secure.*

We provide the proof in Appendix B.

Invisibility. In this paragraph, we prove that IND-PCA secure cryptosystems are mandatory and enough to achieve INV1-CMA secure undeniable signatures. To prove this assertion, we proceed as follows. We first show that the INV1-CMA security of the resulting signatures cannot rest on the NM-CPA security of the underlying cryptosystem. We do this by means of an efficient *meta-reduction* relating such a reduction (the algorithm reducing NM-CPA breaking the underlying cryptosystem to INV1-CMA breaking the construction) to the NM-CPA security of the cryptosystem. Thus, under the assumption that the cryptosystem is NM-CPA secure, the meta reduction forbids the existence of such a reduction. In case the cryptosystem is not NM-CPA secure, such a reduction will be useless. This result will rule out automatically all the other notions that are weaker than NM-CPA, namely, OW-CPA and IND-CPA. Next, we use a similar technique to exclude the OW-CCA notion. The next security notion to be considered is IND-PCA. Luckily, this notion turns out to be sufficient to obtain INV1-CMA secure signatures.

Note that meta-reductions have been successfully used in a number of important cryptographic results, e.g., the result in [7] which proves the impossibility of reducing factoring to the RSA problem, or the results in [38, 36] which show that some well known signatures, which are proven secure in the random oracle, cannot conserve the same security in the standard model. All those impossibility results are partial as they apply only for certain reductions. Our result is in a first stage also partial since it requires the reduction $\mathcal{R}$, trying to attack a certain property of a cryptosystem given by the public key $\Gamma.\mathsf{pk}$, to provide the adversary against the confirmer signature with the confirmer public key $\Gamma.\mathsf{pk}$. We will denote such reductions by *key-preserving* reductions, inheriting the name from a wide and popular class of reductions which supply the adversary with the same public key as its challenge. Such reductions were for instance used in [37] to prove a separation between factoring and IND-CCA-breaking some factoring-based cryptosystems in the standard model. Our restriction to such a class of reductions is not unnatural since, to our best knowledge, all the reductions stemming the security of the generic constructions of confirmer signatures from the security of their underlying components, feed the adversary with the public keys of these components (signature scheme, encryption scheme, commitment scheme). Next, we use simular techniques to [37] to extend our impossibility results to arbitrary reductions.

Lemma 1. *Assume there exists a key-preserving reduction $\mathcal{R}$ that converts an INV1-CMA adversary $\mathcal{A}$ against the above construction to an NM-CPA adversary against the underlying cryptosystem. Then, there exists a meta-reduction $\mathcal{M}$ that NM-CPA breaks the cryptosystem in question.*

Let us first interpret this result. The lemma claims that under the assumption of the underlying cryptosystem being NM-CPA secure, there exists no key-preserving reduction $\mathcal{R}$ that reduces NM-CPA breaking the cryptosystem in question to INV1-CMA breaking the construction, or if there exists such an algorithm, the underlying cryptosystem is not NM-CPA secure, thus rendering such a reduction useless.

Proof. Let $\mathcal{R}$ be a key-preserving reduction that reduces NM-CPA breaking the cryptosystem underlying the construction to INV1-CMA breaking the construction itself. We will construct an algorithm $\mathcal{M}$ that uses $\mathcal{R}$ to NM-CPA break the same cryptosystem by simulating an execution of the INV1-CMA adversary $\mathcal{A}$ against the construction.

Let Γ be the cryptosystem $\mathcal{M}$ is trying to attack. $\mathcal{M}$ launches $\mathcal{R}$ over Γ with the same public key, say $\Gamma.\mathsf{pk}$. $\mathcal{M}$, acting as the INV1-CMA adversary $\mathcal{A}$ against the construction, queries $\mathcal{R}$ on $m_0, m_1 \xleftarrow{R} \{0,1\}^\star$ for confirmer signatures. Then he queries the resulting strings μ_0, μ_1 (corresponding to the confirmer signatures on m_0 and m_1 respectively) for a selective conversion. Let σ_0 and σ_1 be the output (digital) signatures on m_0 and m_1 respectively. At that point, $\mathcal{M}$ inputs $\mathcal{D} = \{\sigma_0, \sigma_1\}$ to his own challenger as a distribution probability from which the plaintexts will be drawn. He gets in response a challenge encryption $\mu^\star$, of either σ_0 or σ_1 under $\Gamma.\mathsf{pk}$, and is asked to produce a ciphertext μ' whose corresponding plaintext is meaningfully related to the decryption of $\mu^\star$. To do this, $\mathcal{M}$ chooses uniformly at random a bit $b \xleftarrow{R} \{0,1\}$. Then, he queries the presumed confirmer signature $\mu^\star$ on m_b for a selective conversion. If the result is different from $\perp$, i.e., $\mu^\star$ is the encryption of σ_b, then $\mathcal{M}$ will output $\Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(\bar{\sigma}_b)$ ($\bar{\sigma}_b$ refers to the bit-complement of the element σ_b) and the relation R: $R(m, m') = (m' = \bar{m})$. Otherwise, he will output $\Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(\bar{\sigma}_{1-b})$ and the same relation R. Finally $\mathcal{M}$ aborts the game (stops simulating an INV1-CMA attacker against the generic construction). □

Lemma 2. *Assume there exists a key-preserving reduction $\mathcal{R}$ that converts an INV1-CMA adversary $\mathcal{A}$ against the above construction to a OW-CCA adversary against the underlying cryptosystem. Then, there exists a meta-reduction $\mathcal{M}$ that OW-CCA breaks the cryptosystem in question.*

As previously, this result claims that under the assumption of the underlying cryptosystem being OW-CCA secure, there exists no key-preserving reduction $\mathcal{R}$ that reduces OW-CCA breaking the cryptosystem in question to INV1-CMA breaking the construction, or if there exists such an algorithm, the underlying cryptosystem is not OW-CCA secure, thus rendering such a reduction useless.

Proof. The proof technique is similar to the one above. Let $\mathcal{R}$ be the key-preserving reduction that reduces OW-CCA breaking the cryptosystem underlying the construction to INV1-CMA breaking the construction itself. We will construct an algorithm $\mathcal{M}$ that uses $\mathcal{R}$ to OW-CCA break the same cryptosystem by simulating an execution of the INV1-CMA adversary $\mathcal{A}$ against the construction.

Let Γ be the cryptosystem $\mathcal{M}$ is trying to attack. $\mathcal{M}$ gets his challenge c and is equipped with a decryption oracle that he can query on all ciphertexts of his choice except of course on the challenge. $\mathcal{M}$ launches $\mathcal{R}$ over Γ with the same public key $\Gamma.\mathsf{pk}$ and the same challenge c. Obviously all decryption queries made by $\mathcal{R}$, which are by definition different from the challenge ciphertext c, can be forwarded to $\mathcal{M}$'s own challenger. At some point, $\mathcal{M}$, acting as an INV1-CMA attacker against the construction, will output two messages m_0, m_1 and gets as response a challenge signature $\mu^\star$ which he is required to tell to which message it corresponds. With overwhelming probability, $\mu^\star \neq c$, in fact, the challenge c is not the encryption of a certain σ such that σ is a valid (digital) signature on the message m_0 or the message m_1. Therefore, $\mathcal{M}$ queries his own challenger for the decryption of $\mu^\star$ (he can issue such a query since it is different from the challenge ciphertext). He checks whether the result, say σ, is a valid (digital) signature on $m^\star$. Then, he will simply output the result of this verification. Finally, when $\mathcal{R}$ outputs his answer, decryption of the ciphertext c, $\mathcal{M}$ will simply forward this result to his challenger.

□

Theorem 2. *The cryptosystem underlying the above construction must be at least IND-PCA secure, in case the considered reduction is key-preserving, in order to achieve INV1-CMA secure signatures.*

Proof. We proceed in this proof with elimination. Lemma 1 rules out the notions NM-CPA and thus the notions IND-CPA and OW-CPA. Moreover Lemma 2 rules out OW-CCA and thus OW-PCA (and also OW-CPA). Thus, the next notion to be considered is IND-PCA. □

Remark 1. The above theorem is only valid when the considered notions are those obtained from pairing a security goal GOAL $\in \{\text{OW}, \text{IND}, \text{NM}\}$ and an attack model ATK $\in \{\text{CPA}, \text{PCA}, \text{CCA}\}$. Presence of other notions will require an additional study, however, Lemmas 1 and 2 will be always of use when there exists a relation between these new notions and the notions OW-CCA, NM-CPA.

Remark 2. One trivial way to exclude the notions OW-CPA, IND-CPA and OW-PCA is to remark that ElGamal's encryption meets all the previously mentioned notions (under the CDH, DDH and GDH assumption resp.) but still cannot be used as an ingredient in the construction. However, these notions still do not exclude homomorphic schemes from the design, which is one goal of the study above. In fact, ElGamal offers the possibility of, given a ciphertext, creating another ciphertext for the same message (multiply the first component by g^r, for some r, and the second one by y^r, where $(\mathsf{sk} = x, \mathsf{pk} = y = g^x)$ is the key pair of the scheme). Now, Let (μ, m_0, m_1) be a challenge to an INV-CMA adversary $\mathcal{A}$. By construction μ is an ElGamal encryption of some σ, which is a digital signature on either m_0 or m_1. By the argument above, $\mathcal{A}$ can create another confirmer signature μ', that is another encryption of σ, and that he can query (w.r.t. m_0 for example) to the selective conversion oracle and then conclude.

To extend the result to arbitrary reductions, we use the same techniques as in [37]. Namely, we first define the notion of *non malleability of a cryptosystem key generator* through the following two games: In **Game 0**, we consider an algorithm $\mathcal{R}$ trying to break a cryptosystem Γ , w.r.t. a public key $\Gamma.\mathsf{pk}$, in the sense of NM-CPA (or OW-CCA) using an adversary $\mathcal{A}$ which solves a problem A, perfectly reducible to OW-CPA breaking the cryptosystem Γ. In this game, $\mathcal{R}$ lunches $\mathcal{A}$ over his own challenge key $\Gamma.\mathsf{pk}$ and some other parameters chosen freely by $\mathcal{R}$. We will denote by $\mathsf{adv}_0(\mathcal{R}^{\mathcal{A}})$ the success probability of $\mathcal{R}$ in such a game, where the probability is taken over the random tapes of both $\mathcal{R}$ and $\mathcal{A}$. We further define $\mathsf{succ}_{\Gamma}^{\mathsf{Game0}}(\mathcal{A}) = \max_{\mathcal{R}} \mathsf{adv}_0(\mathcal{R}^{\mathcal{A}})$ to be the success in **Game 0** of the best reduction $\mathcal{R}$ making the best possible use of the adversary $\mathcal{A}$. In **Game 1**, we consider the same entities as in **Game 0**, with the exception of providing $\mathcal{R}$ with, in addition to $\mathcal{A}$, a OW-CPA oracle (i.e. a decryption oracle corresponding to Γ) that he can query w.r.t. any public key $\Gamma.\mathsf{pk}' \neq \Gamma.\mathsf{pk}$, where $\Gamma.\mathsf{pk}$ is the challenge public key of $\mathcal{R}$. Similarly, we define $\mathsf{adv}_1(\mathcal{R}^{\mathcal{A}})$ to be the success of $\mathcal{R}$ in such a game, and $\mathsf{succ}_{\Gamma}^{\mathsf{Game1}}(\mathcal{A}) = \max_{\mathcal{R}} \mathsf{adv}_0(\mathcal{R}^{\mathcal{A}})$ the success in **Game 1** of the reduction $\mathcal{R}$ making the best possible use of the adversary $\mathcal{A}$ and the of decryption (OW-CPA) oracle.

Definition 1. *A cryptosystem Γ is said to have a non malleable key generator if*
$\Delta = max_{\mathcal{A}} |\mathsf{succ}_{\Gamma}^{\mathsf{Game1}}(\mathcal{A}) - \mathsf{succ}_{\Gamma}^{\mathsf{Game0}}(\mathcal{A})|$ *is negligeable in the security parameter.*

This definition informally means that a cryptosystem has a non malleable key generator if NM-CPA (or OW-CCA) breaking it w.r.t. a key pk is no easier when given access to a decryption (OW-CPA) oracle w.r.t. any public key $\mathsf{pk}' \neq \mathsf{pk}$.

Theorem 3. *If the cryptosystem underlying the above construction has a non malleable key generator, then it must be at least IND-PCA secure in order to achieve INV1-CMA secure confirmer signatures.*

We provide the proof in Appendix C

One can give an informal explanation to the result above as follows. It is well known that constructions obtained from the sign_then_encrypt paradigm are not *strongly unforgeable*. I.e., a polynomial adversary is able to produce, given a valid confirmer signature on a certain message, another valid confirmer signature on the same message without the help of the signer. Indeed, given a valid confirmer signature on a message, an attacker can request its corresponding digital signature from the selective conversion oracle, then he encrypts it under the cryptosystem public key and obtains a new confirmer signature on the same message. Therefore, any reduction $\mathcal{R}$ from the security of the underlying cryptosystem to the invisibility of the construction will need more than a list of records maintaining the queried messages along with the corresponding confirmer and digital signatures. Thus the insufficiency of notions like IND-CPA. In [8], the authors stipulate that the given reduction would need a decryption oracle (of the cryptosystem) in order to handle the queries made by the INV1-CMA attacker $\mathcal{A}$, which makes the invisibility of the construction rest on the IND-CCA security of the cryptosystem. In our work, we remark that the queries made by $\mathcal{A}$ are not completely uncontrolled by $\mathcal{R}$. In fact, they are encryptions of some data already released by $\mathcal{R}$, provided the digital signature scheme is strongly unforgeable, and thus known to him. Therefore, a plaintext checking oracle suffices to handle those queries.

Theorem 4. *The above construction is ($t, \epsilon, q_s, q_v, q_{sc}$)-INV1-CMA secure if the underlying digital signature is (t, ϵ', q_s)-SEUF-CMA secure and the underlying cryptosystem is ($t + q_s q_{sc}(q_{sc} + q_v), \epsilon \cdot (1 - \epsilon')^{(q_{sc}+q_v)}, q_{sc}(q_{sc} + q_v)$)-IND-PCA secure.*

The proof is provided in Appendix D.

Unfortunately, requiring the encryption scheme to be at least IND-PCA secure seems to impact negatively the efficiency of the construction as it excludes homomorphic schemes from use (a homomorphic cryptosystem cannot be IND-PCA secure). In fact, such schemes can be (as we will show later in this document) efficient decryption verifiable, i.e., they accept efficient ZK proofs of knowledge of the decryption of a given ciphertext. In the next section, we discuss an attempt to circumvent this problem.

4 Efficient KEM/DEM-based Constructions

One attempt to circumvent the problem of *strong forgeability* of constructions obtained from the plain "encryption of a signature" paradigm can be achieved by binding the digital signature to its encryption. In this way, from a digital signature σ and a message m, an adversary cannot create a new confirmer signature on m by just reencrypting σ. In fact, σ forms a digital signature on m and some data, say c, which uniquely defines the confirmer signature on m. Moreover, this data c has to be public in order to issue the confirmSign/confirmation/denial protocols. Such an idea has been implemented in [14] in the undeniable signature framework, using the KEM/DEM paradigm; in fact, given a message m, one first fixes the session key k and its encapsulation c, then generates a digital signature σ on the "augmented" message $m\|c$, finally encrypts σ using k and outputs the result as an undeniable signature on m.

In this section, we propose a recast of this construction in the CDCS framework. We also allow more flexibility without compromising the overall security by encrypting only one part of the signature and leaving out the other part, provided it does not reveal information about the key or the message. Moreover, we demonstrate the efficiency of the resulting construction by explicitly describing its confirmSign/confirmation/denial protocols when the underlying components belong to a wide class of encryption and digital signature schemes. Interestingly, the class of digital signatures we consider has been already used in a recent proposal [41] as an ingredient for a generic construction of designated-verifier signatures. Finally, we conclude with a comparison with the existing generic constructions.

4.1 The construction

Let Σ be a digital signature scheme given by $\Sigma.\mathsf{keygen}$ which generates a key pair ($\Sigma.\mathsf{sk}$, $\Sigma.\mathsf{pk}$), $\Sigma.\mathsf{sign}$ and $\Sigma.\mathsf{verify}$. Let furthermore $\mathcal{K}$ be a KEM given by $\mathcal{K}.\mathsf{keygen}$ which generates a key pair ($\mathcal{K}.\mathsf{pk}$, $\mathcal{K}.\mathsf{sk}$), $\mathcal{K}.\mathsf{encap}$ and $\mathcal{K}.\mathsf{decap}$. Finally, we consider a DEM $\mathcal{D}$ given by $\mathcal{D}.\mathsf{encrypt}$ and $\mathcal{D}.\mathsf{decrypt}$.

Without loss of generality, we consider that a digital signature σ generated using Σ on a message m, can written on the form $\sigma = (s, r)$ where r reveals no information about m nor about $(\Sigma.\mathsf{sk}, \Sigma.\mathsf{pk})$. I.e., there exists an algorithm that inputs a message m and a key pair $(\Sigma.\mathsf{sk}, \Sigma.\mathsf{pk})$ and outputs a string indistinguishable from r, where the probability is taken over the message and the key pair spaces considered by Σ. Note that every signature scheme produces signatures of the given form, since a signature can be always written as the concatenation of itself and the empty string (the message-key-independent part). We assume that s belongs to the message space of $\mathcal{D}$.

Let $m \in \{0, 1\}^\star$ be a message, we propose the following recast of the construction in [14].

Key generation. Call $\Sigma.\mathsf{keygen}$ and $\mathcal{K}.\mathsf{keygen}$ to generate $\Sigma.\mathsf{sk}$, $\Sigma.\mathsf{pk}$, $\mathcal{K}.\mathsf{pk}$ and $\mathcal{K}.\mathsf{sk}$ respectively. Set the signer key pair to $(\Sigma.\mathsf{sk}, \Sigma.\mathsf{pk})$ and the confirmer key pair to $(\mathcal{K}.\mathsf{sk}, \mathcal{K}.\mathsf{pk})$.

ConfirmSign. Fix a key k together with its encapsulation e. Then compute a (digital) signature $\sigma = \Sigma.\mathsf{sign}_{\Sigma.\mathsf{sk}}(m\|e) = (s, r)$ on $m\|e$. Finally, output $\mu = (e, \mathcal{D}.\mathsf{encrypt}_k(s), r)$ and prove the knowledge of s, decryption of $(e, \mathcal{D}.\mathsf{encrypt}_k(s))$, which satisfies together with r $\Sigma.\mathsf{verify}$. This proof is possible because the signer knows k and (s, r), and the last assertion defines an NP language which accepts a ZK proof system.

Confirmation/Denial protocol. To confirm (deny) a purported signature $\mu = (\mu_1, \mu_2, \mu_3)$, issued on a certain message m, the confirmer first computes $k = \mathcal{K}.\mathsf{decap}_{\mathcal{K}.\mathsf{sk}}(\mu_1)$ then calls $\Sigma.\mathsf{verify}$ on $(\mathcal{D}.\mathsf{decrypt}_k(\mu_2), \mu_3)$ and $m\|\mu_1$ using $\Sigma.\mathsf{pk}$. According to the result, the signer issues a ZK proof of knowledge of the decryption of (μ_1, μ_2) that, together with μ_3, passes (does not pass) the verification algorithm $\Sigma.\mathsf{verify}$. Again this proof is possible because the given assertions are either NP or co-NP statements and therefore accept a ZK proof system.

Selective conversion. To convert a given signature $\mu = (\mu_1, \mu_2, \mu_3)$ issued on a certain message m, the signer first checks its validity. In case it is valid, the signer computes $k = \mathcal{K}.\mathsf{decap}_{\mathcal{K}.\mathsf{sk}}(\mu_1)$ and outputs $(\mathcal{D}.\mathsf{decrypt}_k(\mu_2), \mu_3)$ otherwise he outputs $\perp$.

Theorem 5. *The above construction is (t, ϵ, q_s)-EUF-CMA secure if the underlying digital signature scheme is (t, ϵ, q_s)-EUF-CMA secure.*

Theorem 6. *The proposed construction is $(t, \epsilon, q_s, q_v, q_{sc})$-INV2-CMA secure if it uses a (t, ϵ', q_s)-EUF-CMA secure digital signature, an INV-OT secure DEM and a $(t+q_s(q_v+q_{sc}), \epsilon\cdot(1-\epsilon')^{q_v+q_{sc}})$-IND-CPA secure KEM.*

The proofs are similar to those provided in [28]. Note that the strong unforgeability of the underlying signature scheme is not needed here to achieve invisibility. In fact, if the adversary can come up with another digital signature σ' on a given $m\|c$, there is just one way to create the corresponding confirmer signature, namely, encrypt it using $k = \mathcal{K}.\mathsf{decap}(c)$. Therefore, the reduction is able to handle a query requesting the confirmation/denial or selective conversion of such a signature by just maintaining a list of the queried messages, the issued confirmer signatures and their corresponding digital signatures.

4.2 Efficient Instantiations using Certain Signatures and Cryptosystems

In this paragraph, we define the classes of signatures/cryptosystems that yield efficient instantiations of the construction defined earlier in this section. The class of digital signatures we consider is very similar to the one defined by [41] in the context of designated verifier signatures, whereas the class of considered cryptosystems spotlights the importance of homomorphic encryption in the framework.

Definition 2. ***(The class $\mathbb{S}$ of signatures)*** *$\mathbb{S}$ is the set of all digital signatures for which there exists a pair of algorithms,* Convert *and* Retrieve, *where* Convert *inputs a public key* pk, *a message m, and a valid signature σ on m (according to* pk*) and outputs the pair (s, r) such that:*

1. *there exists an algorithm that inputs a public key* pk *from the key space and a message m from the message space, and outputs a string statistically indistinguishable from r.*
2. *there exists an algorithm* Compute *that on the input the public key* pk*, the message m and r, computes a description of a* one-way function $f : (\mathbb{G}, *) \to (\mathbb{H}, \circ_s)$*:*
   - *where $(\mathbb{G}, *)$ is group and $\mathbb{H}$ is a set equipped with the binary operation $\circ_s$,*
   - *$\forall S, S' \in \mathbb{G}$: $f(S * S') = f(S) \circ_s f(S')$.*

   and an $I \in \mathbb{H}$, such that $f(s) = I$.

and Retrieve *is an algorithm that inputs* pk, *m and the correctly converted pair (s, r) and retrieves the signature σ on m.*

The class $\mathbb{S}$ differs from the class $\mathbb{C}$, introduced in [41], in the condition required for the one way function f. In fact, in our description of $\mathbb{S}$, the function f should satisfy a homomorphic property, whereas in the class $\mathbb{C}$, f should only possess an efficient *Σ protocol* for proving knowledge of a preimage of a value in its range. We show in Theorem 7 that signatures in $\mathbb{S}$ accept also efficient *Σ protocols* for proving knowledge of preimages, and thus belong to the class $\mathbb{C}$. Conversely, one can claim that signatures in $\mathbb{C}$ are also in $\mathbb{S}$, at least from a practical point of view, since it is not known how to achieve efficient Σ protocols for proving knowledge of preimages of f without having the latter item satisfy some homomorphic properties. It is worth noting that similar to the classes $\mathbb{S}$ and $\mathbb{C}$ is the class of signatures introduced in [23], where the condition of having an efficient Σ protocol for proving knowledge of preimages is weakened to having only a *witness hiding* proof of knowledge. Again, although this is a weaker assumption on f, all illustrations of signatures in this wider class happen to be also in $\mathbb{C}$ and $\mathbb{S}$. Our resort to specify the homomorphic property on f will be justified later when describing the confirmation/denial protocols of the resulting construction. In fact, these protocols are parallel composition of Σ protocols and therefore need a careful study as it is known that zero knowledge is not close under concurrent composition. Finally, the class $\mathbb{S}$ encompasses most proposals that were suggested so far, RSA-FDH [3], Schnorr [42], GHR [18], Modified ElGamal [39], Cramer-Shoup [13], Camenisch-Lysyanskaya-02 [24] and most pairing-based signatures such as [6, 25, 4, 46, 45].

1. The prover chooses $s' \xleftarrow{R} \mathbb{G}$, computes and sends $t_1 = I \circ_s f(s')$ to the verifier.
2. The verifier chooses $c \xleftarrow{R} \{0, 1\}$ and sends it to the prover.
3. If $b = 0$, the prover sends s'.
   Otherwise, he sends $s * s'$.
4. If $b = 0$, the verifier checks that t_1 is computed as in Step 1.
   Otherwise, he verifier accepts if $f(s * s') = t_1$.

1

Fig. 1. Proof system for membership to the language $\{s\colon f(s) = I\}$ Common input: I and Private input : s

Theorem 7. *The protocol depicted in Figure 1 is an efficient Σ protocol, ZK close under parallel composition, for proving knowledge of preimages of the function f described in Definition 2.*

The proof will be given in Appendix E.1.

Definition 3. ***(The class $\mathbb{E}$ of cryptosystems)*** *$\mathbb{E}$ is the set of encryption schemes Γ,* obtained from the KEM/DEM paradigm, *that have the following properties:*

1. *The message space is a group $\mathcal{M} = (\mathbb{G}, *)$ and the ciphertext space $\mathcal{C}$ is a set equipped with a binary operation $\circ_e$.*
2. *Let $m \in \mathcal{M}$ be a message and c its encryption with respect to a key* pk*. On the common input m and c, there exists an efficient zero knowledge proof of m being the decryption of c with respect to* pk*. The private input of the prover is either the private key* sk*, corresponding to* pk *or the randomness used to encrypt m in c (the randomness which is input to the KEM encapsulation algorithm).*
3. *$\forall m, m' \in \mathcal{M}$, $\forall \mathsf{pk}$: $\Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m * m') = \Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m) \circ_e \Gamma.\mathsf{encrypt}_{pk}(m')$. Moreover, given the randomness used to encrypt m in $\Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m)$ and m' in $\Gamma.\mathsf{encrypt}_{pk}(m')$, one can deduce (using only the public parameters) the randomness used to encrypt $m * m'$ in $\Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m * m')$.*

Examples of cryptosystems in the above class are ElGamal's encryption [15], or the cryptosystem defined in [5] which uses the linear Diffie-Hellman KEM. In fact, both cryptosystems are homomorphic and possess an efficient protocol for proving that a ciphertext decrypts to a given plaintext: the proof of equality of two discrete logarithms [11]. Paillier's [35] cryptosystem cannot be viewed as an instance

```
1. The prover chooses $s' \xleftarrow{R} \mathbb{G}$, computes and sends $t_2 = \Gamma.\mathsf{encrypt}(s') \circ_e (c, s_k)$ to the verifier
2. The verifier chooses $c \xleftarrow{R} \{0,1\}$ and sends it to the signer.
3. If $b = 0$, the prover sends $s'$ and the randomness used to encrypt $\Gamma.\mathsf{encrypt}(s')$.
   Otherwise, he sends $s' * s$ and proves that $t_2$ is an encryption of $s' * s$.          l
4. If $b = 0$, the verifier checks that $t_2$ is computed as in Step 1.
   Otherwise, he checks the proof of decryption of $t_2$:
      It it fails, he rejects the proof.
```

Fig. 2. Proof system for membership to the language $\{(e, s_k) \colon \exists m : m = \Gamma.\mathsf{decrypt}(e, s_k)\}$ Common input: $(e, s_k, \Gamma.\mathsf{pk})$ and Private input: $\Gamma.\mathsf{sk}$ or randomness encrypting m in (e, s_k)

of this class as it is not based on the KEM/DEM paradigm, however in Appendix E.2, we provide a modified variant which belongs to the class $\mathbb{E}$ and thus is suitable for use in the construction.

Theorem 8. *Let Γ be a cryptosystem from the above class $\mathbb{E}$. Let furthermore c be an encryption of some message under some public* pk. *The protocol depicted in Figure 2 is an efficient Σ protocol, ZK close under parallel composition, for proving knowledge of the decryption of c.*

The proof is similar to the one of Theorem 7. □

The confirmation/denial protocol We combine an EUF-CMA secure signature scheme $\Sigma \in \mathbb{S}$ and a cryptosystem $\Gamma \in \mathbb{E}$, where the underlying KEM $\mathcal{K}$ and DEM $\mathcal{D}$ are IND-CPA and INV-OT secure respectively, in the way descried in Section 4. Namely, we first compute an encapsulation e together with its corresponding key k. Then compute a signature σ on the message to be signed concatenated with e. Finally convert σ to (s, r) using the Convert algorithm described in Definition 2 and encrypt s using k. The resulting confirmer signature is $(e, \mathcal{D}.\mathsf{encrypt}_k(s), r)$. We describe in Figure 3 the confirmation/denial protocols corresponding to the resulting construction. Note that the confirmation protocol can be also run by the signer who wishes to confirm the validity of a just generated signature.

```
1. The prover and verifier, given the public input, compute $I$ as defined in Definition 2.
2. The prover chooses $s' \xleftarrow{R} \mathbb{G}$, computes and sends $t_1 = f(s') \circ_s I$ and
$t_2 = \Gamma.\mathsf{encrypt}(s') \circ_e (e, s_k)$ to the verifier
3. The verifier chooses $c \xleftarrow{R} \{0,1\}$ and sends it to the prover.
4. If $b = 0$, the prover sends $s'$ and the randomness used to encrypt $s'$ in $\Gamma.\mathsf{encrypt}(s')$.
   Otherwise, he sends $s' * s$ and proves that $t_2$ is an encryption of $s' * s$.          l
5. If $b = 0$, the verifier checks that $t_1$ and $t_2$ are computed as in Step 1.
   Otherwise, he checks the proof of decryption of $t_2$:
      It it fails, he rejects the proof.
      Otherwise:
          If the prover is confirming the signature, the verifier accepts if $f(s' * s) = t_1$.
          If the prover is denying the given signature, the verifier accepts the proof if $f(s' * s) \neq t_1$.
```

Fig. 3. Proof system for membership (non membership) to the language $\{(e, s_k, r) \colon \exists s : s = \Gamma.\mathsf{decrypt}(e, s_k) \wedge \Sigma.\mathsf{verify}(\mathsf{Retrieve}(s, r), m \| e) = (\neq) 1\}$ Common input: $(e, s_k, r, \Sigma.\mathsf{pk}, \Gamma.\mathsf{pk})$ and Private input: $\Gamma.\mathsf{sk}$ or randomness encrypting s in (e, s_k)

Remark 3. The prover in Figure 3 is either the confirmer of the signature (e, s_k, r) who can run the above protocols with the knowledge of his private key, or the signer who wishes to confirm the validity of a just generated signature. In fact, with the knowledge of the randomness used to encrypt s in (e, s_k), where

(s, r) is the converted pair obtained from $\sigma = \Sigma.\mathsf{sign}(m\|e)$, the signer can issue the above confirmation protocol thanks to the properties satisfied by Γ.

Theorem 9. *The confirmation protocol (run either by the signer on a just generated signature or by the confirmer on any signature) described in Figure 3 is a Σ protocol which is ZK close under parallel composition.*

Theorem 10. *The denial protocol described in Figure 3 is a Σ protocol, ZK close under parallel composition, under the assumption of the underlying cryptosystem being IND-CPA-secure.*

The proofs of both theorems are given in Appendices E.3 and E.4 respectively.

4.3 Comparisons and possible extentions

sign_then_encrypt variants. The construction presented in this section improves the plain paradigm [8] as it weakens the assumption on the underlying cryptosystem from being IND-CCA secure to only being IND-CPA secure. This impacts positively the efficiency of the construction from many sides. In fact, the resulting signature is shorter and its generation cost is smaller, since IND-CPA cryptosystems are simpler and allow faster encryption and shorter ciphertexts than IND-CCA ones. An illustration is given by ElGamal's encryption and its IND-CCA variant, namely Cramer-Shoup's encryption where the ciphertexts are at least twice longer than ElGamal's ciphertexts. Also, there is a multiplicative factor of at least two in favor of ElGamal's encryption/decryption cost. Moreover, the confirmation/denial protocols are rendered more efficient by the allowance of homomorphic cryptosystems as shown in 4.2. Such cryptosystems were not possible to use before, since a homomorphic scheme can never attain the IND-CCA security. Besides, even when the IND-CCA cryptosystem is decryption verifiable, e.g., Cramer-Shoup or the IND-CCA variant of Paillier's encryption [9], the involved protocols are much more expensive than the ones corresponding to their IND-CPA variant: in case of ElGamal, this protocol amounts to a proof of equality of two discrete logarithms, and in case of our modified variant of Paillier (Appendix E.2), this protocol comes to a proof of knowledge of an N-th root. The construction achieves also better performances than the proposal of [23], where the confirmer signature comprises k commitments and $2k$ IND-CCA encryptions, where k is the number of rounds used in the confirmation protocol. Moreover, the denial protocol presented in [23] suffers the resort to proofs of general NP statements (where the considered encryption is IND-CCA). Finally, the resulting signatures are not invisible.

Commitment-based constructions. Our construction does not use ROM, unlike the constructions in [30, 44]. Moreover, it enjoys the strongest notion of invisibility (INV2-CMA) which captures both invisibility as defined in [8], and anonymity as defined in [17]. As mentioned in subsection 2.2, anonymity can be an important requirement for confirmer signatures in some settings. Unfortunately, many of the efficient generic constructions are not anonymous. In fact, constructions like [30, 19, 44] have a confirmer signature containing a commitment on the message to be signed and a valid digital signature on this commitment. Therefore, such constructions leak always a part of the signing key, namely the public key of the underlying digital signature. More precisely, an anonymity attacker $\mathcal{A}$, will get two public keys and a confirmer signature on a given message and has to tell the key under which the confirmer signature was created. To answer such a challenge, $\mathcal{A}$ will simply check the validity of the digital signature on the commitment (both are part of the confirmer signature) with regard to one public key (the confirmer signature public key includes the public key of the underlying digital signature). The result of such a verification is sufficient for $\mathcal{A}$ to conclude in case the two confirmer public keys do not share the same public key for the digital signature scheme.

References

1. A. K. Lenstra, *Generating RSA Moduli with a Predetermined Portion*, ASIACRYPT (K. Ohta and D. Pei, eds.), LNCS, vol. 1514, Springer, 1998, pp. 1–10.
2. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway, *Relations Among Notions of Security for Public-Key Encryption Schemes.*, Advances in Cryptology - CRYPTO'98 (H. Krawczyk, ed.), LNCS, vol. 1462, Springer, 1998, pp. 26–45.
3. M. Bellare and P. Rogaway, *The Exact Security of Digital Signatures: How to Sign with RSA and Rabin.*, in Maurer [29], pp. 399–416.
4. D. Boneh and X. Boyen, *Short Signatures Without Random Oracles.*, Advances in Cryptology - EUROCRYPT 2004 (C. Cachin and J. Camenisch, eds.), LNCS, vol. 3027, Springer, 2004, pp. 56–73.
5. D. Boneh, X. Boyen, and H. Shacham, *Short Group Signatures.*, in Franklin [16], pp. 41–55.
6. D. Boneh, B. Lynn, and H. Shacham, *Short Signatures from the Weil Pairing.*, J. Cryptology **17** (2004), no. 4, 297–319.
7. D. Boneh and R. Venkatesan, *Breaking RSA May Not Be Equivalent to Factoring.*, in Nyberg [31], pp. 59–71.
8. J. Camenisch and M. Michels, *Confirmer Signature Schemes Secure against Adaptative Adversaries.*, Advances in Cryptology - EUROCRYPT 2000 (B. Preneel, ed.), LNCS, vol. 1807, Springer, 2000, pp. 243–258.
9. J. Camenisch and V. Shoup, *Practical Verifiable Encryption and Decryption of Discrete Logarithms.*, Advances in Cryptology - CRYPTO 2003 (D. Boneh, ed.), LNCS, vol. 2729, Springer, 2003, pp. 126–144.
10. D. Chaum, *Designated Confirmer Signatures.*, Advances in Cryptology - EUROCRYPT'94 (A. De Santis, ed.), LNCS, vol. 950, Springer, 1995, pp. 86–91.
11. D. Chaum and T. P. Pedersen, *Wallet Databases with Observers.*, Advances in Cryptology - CRYPTO'92 (E. F. Brickell, ed.), LNCS, vol. 740, Springer, 1993, pp. 89–105.
12. D. Chaum and H. van Antwerpen, *Undeniable Signatures.*, Advances in Cryptology - CRYPTO'89 (G. Brassard, ed.), LNCS, vol. 435, Springer, 1990, pp. 212–216.
13. R. Cramer and V. Shoup, *Signature schemes based on the strong RSA assumption.*, ACM Trans. Inf. Syst. Secur. **3** (2000), no. 3, 161–185.
14. L. El Aimani, *Toward a Generic Construction of Universally Convertible Undeniable Signatures from Pairing-Based Signatures*, Progress in Cryptology - INDOCRYPT 2008 (D. Roy Chowdhury, V. Rijmen, and A. Das, eds.), LNCS, vol. 5365, Springer, 2008, pp. 145–157.
15. T. El Gamal, *A Public Key Cryptosystem and a Signature Scheme based on Discrete Logarithms.*, IEEE Trans. Inf. Theory **31** (1985), 469–472.
16. M. K. Franklin (ed.), *Advances in Cryptology - CRYPTO 2004, 24th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, LNCS, vol. 3152, Springer, 2004.
17. S. D. Galbraith and W. Mao, *Invisibility and Anonymity of Undeniable and Confirmer Signatures.*, Topics in Cryptology - CT-RSA 2003 (M. Joye, ed.), LNCS, vol. 2612, Springer, 2003, pp. 80–97.
18. R. Gennaro, S. Halevi, and T. Rabin, *Secure Hash-and-Sign Signatures Without the Random Oracle.*, in Stern [43], pp. 397–416.
19. C. Gentry, D. Molnar, and Z. Ramzan, *Efficient Designated Confirmer Signatures Without Random Oracles or General Zero-Knowledge Proofs*, in Roy [40], pp. 662–681.
20. O. Goldreich, *Foundations of cryptography. Basic Tools.*, Cambridge University Press., 2001.
21. Oded Goldreich, Silvio Micali, and Avi Wigderson, *How to Prove all NP-Statements in Zero-Knowledge, and a Methodology of Cryptographic Protocol Design*, CRYPTO (A. M. Odlyzko, ed.), LNCS, vol. 263, Springer, 1986, pp. 171–185.
22. S. Goldwasser, S. Micali, and R. L. Rivest, *A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks.*, SIAM J. Comput. **17** (1988), no. 2, 281–308.
23. S. Goldwasser and E. Waisbard, *Transformation of Digital Signature Schemes into Designated Confirmer Signature Schemes.*, Theory of Cryptography, TCC 2004 (M. Naor, ed.), LNCS, vol. 2951, Springer, 2004, pp. 77–100.
24. J. Camenisch and A. Lysyanskaya, *Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials*, CRYPTO (M. Yung, ed.), LNCS, vol. 2442, Springer, 2002, pp. 61–76.
25. ______, *Signature Schemes and Anonymous Credentials from Bilinear Maps*, in Franklin [16], pp. 56–72.
26. J. Herranz, D. Hofheinz, and E. Kiltz, *KEM/DEM: Necessary and Sufficient Conditions for secure Hybrid Encryption*, Available at http://eprint.iacr.org/2006/265.pdf, August 2006.
27. M. Jakobsson, K. Sako, and R. Impagliazzo, *Designated Verifier Proofs and Their Applications.*, in Maurer [29], pp. 143–154.
28. L. El Aimani, *Toward a Generic Construction of Convertible Undeniable Signatures from Pairing-Based Signatures*, Cryptology ePrint Archive, Report 2009/362, 2009, `http://eprint.iacr.org/`.
29. U. M. Maurer (ed.), *Advances in Cryptology - EUROCRYPT'96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, LNCS, vol. 1070, Springer, 1996.
30. M. Michels and M. Stadler, *Generic Constructions for Secure and Efficient Confirmer Signature Schemes.*, in Nyberg [31], pp. 406–421.
31. K. Nyberg (ed.), *Advances in Cryptology - EUROCRYPT'98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, LNCS, vol. 1403, Springer, 1998.

32. T. Okamoto, *Designated Confirmer Signatures and Public-Key Encryption are Equivalent.*, Advances in Cryptology - CRYPTO'94 (Y. Desmedt, ed.), LNCS, vol. 839, Springer, 1994, pp. 61–74.
33. T. Okamoto and D. Pointcheval, *The Gap-Problems: a New Class of Problems for the Security of Cryptographic Schemes.*, 4th International Workshop on Practice and Theory in Public Key Cryptography, PKC 2001 (K. Kim, ed.), LNCS, vol. 1992, Springer, 2001, pp. 104–118.
34. P. Le Trieu and K. Kurosawa and W. Ogata, *New RSA-Based (Selectively) Convertible Undeniable Signature Schemes*, AFRICACRYPT 2009 (B. Preneel, ed.), LNCS, vol. 5580, Springer, 2009, pp. 116–134.
35. P. Paillier, *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes*, in Stern [43], pp. 223–238.
36. ______, *Impossibility Proofs for RSA Signatures in the Standard Model*, CT-RSA (M. Abe, ed.), LNCS, vol. 4377, Springer, 2007, pp. 31–48.
37. P. Paillier and J. Villar, *Trading One-Wayness Against Chosen-Ciphertext Security in Factoring-Based Encryption*, ASIACRYPT (X. Lai and K. Chen, eds.), LNCS, vol. 4284, Springer, 2006, pp. 252–266.
38. P. Paillier and D. Vergnaud, *Discrete-Log Based Signatures May Not Be Equivalent to Discrete-Log.*, in Roy [40], pp. 1–20.
39. D. Pointcheval and J. Stern, *Security Arguments for Digital Signatures and Blind Signatures.*, J. Cryptology **13** (2000), no. 3, 361–396.
40. B. Roy (ed.), *Advances in Cryptology* - ASIACRYPT *2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, Taj Coromandel, Chennai, India December 4-8, 2005, Proceedings*, LNCS, vol. 3788, Springer, 2005.
41. S. F. Shahandashti and R. Safavi-Naini, *Construction of Universal Designated-Verifier Signatures and Identity-Based Signatures from Standard Signatures*, PKC 2008 (R. Cramer, ed.), LNCS, vol. 4939, Springer, 2008, pp. 121–140.
42. C. P. Schnorr, *Efficient signature generation by smart cards.*, J. Cryptology **4** (1991), no. 3, 161–174.
43. J. Stern (ed.), *Advances in Cryptology* - EUROCRYPT*'99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, LNCS, vol. 1592, Springer, 1999.
44. G. Wang, J Baek, D. S. Wong, and F. Bao, *On the Generic and Efficient Constructions of Secure Designated Confirmer Signatures*, PKC 2007 (T. Okamoto and X. Wang, eds.), LNCS, vol. 4450, Springer, 2007, pp. 43–60.
45. B. Waters, *Efficient Identity-Based Encryption Without Random Oracles.*, Advances in Cryptology - EUROCRYPT 2005 (R. Cramer, ed.), LNCS, vol. 3494, Springer, 2005, pp. 114–127.
46. F. Zhang, R. Safavi-Naini, and W. Susilo, *An Efficient Signature Scheme from Bilinear Pairings and Its Applications.*, 7th International Workshop on Practice and Theory in Public Key Cryptography, PKC 2004 (F. Bao, R. H. Deng, and J. Zhou, eds.), LNCS, vol. 2947, Springer, 2004, pp. 277–290.

A Preliminaries

A.1 Digital signatures

A signature scheme Σ comprises three algorithms, namely the key generation algorithm keygen, the signing algorithm sign, and the verification algorithm verify. The standard security notion for a signature scheme is existential unforgeability under chosen message attacks (EUF-CMA), which was introduced in [22]. Informally, this notion refers to the hardness of, given a signing oracle, producing a valid pair of message and corresponding signature such that message has not been queried to the signing oracle. There exists also the stronger notion, SEUF-CMA (strong existential unforgeability under chosen message attack), which allows the adversary to produce a forgery on a previously queried message, however the corresponding signature must not be obtained from the signing oracle.

A.2 Public key encryption schemes

A public key encryption (PKE) scheme consists of the key generation algorithm keygen, the encryption algorithm encrypt and the decryption algorithm decrypt. The typical *security goals* a cryptosystem should attain are: one-wayness (OW) which corresponds the difficulty of recovering the plaintext from a ciphertext, indistinguishability (IND) which refers to the hardness of distinguishing ciphertexts based on the messages they encrypt, and finally non-Malleability (NM) which corresponds to the hardness of deriving from a given ciphertext another ciphertext such that the underlying plaintexts are meaningfully related. Conversely, the typical *attack models* an adversary against an encryption scheme is allowed to are: Chosen Plaintext Attack (CPA) where the adversary can encrypt any message of his choice. This is inevitable in public key settings, Plaintext Checking Attack (PCA) in which the adversary is allowed to query an oracle on pairs (m, c) and gets answers whether m is really encrypted in c or not, and finally Chosen Ciphertext Attack (CCA) where the adversary is allowed to query a decryption oracle. Pairing the mentioned goals with these attack models yields nine *security notions*: GOAL-ATK for GOAL $\in \{\text{OW}, \text{IND}, \text{NM}\}$ and ATK $\in \{\text{CPA}, \text{PCA}, \text{CCA}\}$. We refer to [2] for the formal definitions of these notions as well as for the relations they satisfy.

A.3 Key/Data encapsulation mechanisms (KEM/DEMs)

A KEM comprises three algorithms: the key generation algorithm keygen, the encapsulation algorithm encap and the decapsulation algorithm decap. The typical security goals that a KEM should satisfy are similar to the ones defined for encryption schemes. Similarly, when conjoined with the three attack models CPA, PCA and CCA, they yield nine security notions whose definitions follow word-for-word from the definitions of the encryption schemes notions. A DEM is simply a secret key encryption scheme given by the same algorithms forming a cryptosystem (PKE). KEMs could be efficiently combined with DEMs to build secure encryption schemes. This paradigm is called the Hybrid encryption paradigm and we refer to [26] for the necessary and sufficient conditions on the KEMs and the DEMs in order to obtain a certain level of security for the resulting hybrid encryption scheme. For instance, to obtain an IND-CPA secure cryptosystem, it suffices to combine an IND-CPA secure KEM and *indistinguishable under a one time attack (IND-OT)* DEM. Finally, we need to define a further notion for DEMs:

Definition 4. *A DEM is said to be* invisible under a one time attack (INV-OT) *if no polynomial adversary $\mathcal{A}$ wins the following game (running in three phases) with non negligeable probability.* ***Phase 1.*** *The challenger runs the algorithm $\mathcal{D}.\mathsf{keygen}$ to obtain a key $\mathcal{D}.\mathsf{sk}$.* ***Challenge.*** *The adversary outputs eventually a message $m^\star$. The challenger picks uniformly at random a bit b from $\{0, 1\}$. If $b = 0$, he encrypts $m^\star$, in $e^\star$, under $\mathcal{D}.\mathsf{sk}$. Otherwise, he chooses a string uniformly at random from the ciphertext space.* ***Phase 2.*** *$\mathcal{A}$ outputs a bit b', representing his guess of $e^\star$ being the encryption of $m^\star$ and wins the game if $b = b'$. We define $\mathcal{A}$'s advantage as $\mathsf{adv}(\mathcal{A}) = |\Pr[b = b'] - \frac{1}{2}|$, where the probability is taken over the random choices of the adversary $\mathcal{A}$ and the challenger.*

A.4 Σ protocols

A Σ protocol is an argument of knowledge which is complete, sound and Honest Verifier Zero Knowledge (HVZN), which is close under parallel composition. We refer to [20] for more information.

B Proof of Theorem 1

Proof. The **If direction** has been already proved in [8]. We prove now the other direction. Let $(m^\star, \sigma^\star)$ be an existential forgery against the digital signature scheme. One can derive a forgery against the confirmer signature by simply encrypting the signature $\sigma^\star$ using the public key of the confirmer. Simulation of the attacker's environment is easy; the reduction $\mathcal{R}$ (EUF-CMA attacker against the confirmer signature) will forward the appropriate parameters (those concerning the underlying digital signature) to the EUF-CMA attacker against the underlying signature scheme, denoted $\mathcal{A}$. For a signature query on a message m, R will first request his challenger for a confirmer signature μ that he decrypts using the universal trapdoor (the cryptosystem private key) in σ. σ forms the result output to $\mathcal{A}$. □

C Proof of Theorem 3

The proof is similar to a combination of Lemma 1 and Theorem 5 in [37]

Proof. We first note that the purpose of **Game 0** is to include all the key-preserving reductions which feed the adversary $\mathcal{A}$ with the same challenge public key in addition to some other parameters. Next we remark that the advantage of the meta-reduction $\mathcal{M}$ in the proof of Lemma 1 (2) is the same as the advantage of any key-preserving reduction $\mathcal{R}$ reducing NM-CPA (OW-CCA) breaking a cryptosystem Γ to breaking the invisibility of a given confirmer signature. For instance, this applies to the reduction making the best use of an invisibility adversary $\mathcal{A}$ against the construction. Therefore we have:

$$\mathsf{succ}_{\Gamma}^{\mathsf{Game0}}(\mathcal{A}) \leq \mathsf{succ}(NM-CPA[\Gamma])$$

where $\mathsf{succ}(NM-CPA[\Gamma])$ is the success of breaking Γ in the NP-CPA sense. We also have $\mathsf{succ}_{\Gamma}^{\mathsf{Game0}}(\mathcal{A}) \leq \mathsf{succ}(OW-CCA[\Gamma])$.

Next, we prove that for any *arbitrary* reduction $\mathcal{R}$ that NM-CPA (OW-CCA) breaks a cryptosystem Γ, given access to an invisibility adversary $\mathcal{A}$ against the construction (of a confirmer signature using Γ), we have

$$\mathsf{adv}(\mathcal{R}) \leq \mathsf{succ}_{\Gamma}^{\mathsf{Game1}}(\mathcal{A})$$

In fact, assume that $\mathcal{R}$ breaks the NM-CPA (OW-CCA) security. We construct an algorithm $\mathcal{M}$ that plays **Game 1** with respect to perfect oracle for $\mathcal{A}$ and succeeds in breaking the NM-CPA (OW-CCA) security of Γ with similar success probability. Algorithm $\mathcal{M}$ gets a challenge w.r.t. a public key pk and launches $\mathcal{R}$ over the same challenge and the same public key. If $\mathcal{R}$ calls $\mathcal{A}$ on pk, then $\mathcal{M}$ will call his own oracle for $\mathcal{A}$. Otherwise, if $\mathcal{R}$ calls $\mathcal{A}$ on $\mathsf{pk}' \neq \mathsf{pk}$, $\mathcal{M}$ will invoke his own decryption oracle for pk' (OW-CPA oracle) to first decrypt the confirmer signature and then check whether the result is a valid digital signature on the message in question. The output of the verification algorithm is sufficient for $\mathcal{M}$ to answer such queries. Finally, when $\mathcal{R}$ outputs the result to $\mathcal{M}$, the latter will output the same result to his own challenger.

Now, Let $\mathcal{R}$ be an arbitrary reduction from NM-CPA (OW-CCA) breaking a cryptosystem Γ, with a non malleable key generator, to INV1-CMA breaking the construction. We have

$$\begin{aligned}\mathsf{adv}(\mathcal{R}) &\leq \mathsf{succ}_{\Gamma}^{\mathsf{Game1}}(\mathcal{A})\\ &\leq \mathsf{succ}_{\Gamma}^{\mathsf{Game0}}(\mathcal{A}) + \Delta\\ &\leq \mathsf{succ}(NM-CPA[\Gamma])(\mathsf{succ}(OW-CCA[\Gamma])) + \Delta\end{aligned}$$

since Δ is negligeable, then under the assumption of Γ being NM-CPA (OW-CCA) secure, the advantage of $\mathcal{R}$ is also negligeable. □

D Proof of Theorem 4

Proof. Let $\mathcal{A}$ be an attacker that $(t, \epsilon, q_s, q_v, q_{sc})$-INV1-CMA breaks the invisibility of the above confirmer signature, believed to be (t, ϵ', q_s)-EUF-CMA secure. We will construct an algorithm $\mathcal{R}$ that IND-PCA breaks the underlying cryptosystem as follows.

Phase 1

- **Key generation.** $\mathcal{R}$ will get the public parameters of the target cryptosystem from his challenger, that are $\Gamma.\mathsf{pk}$, $\Gamma.\mathsf{encrypt}$ and $\Gamma.\mathsf{decrypt}$. Then he will choose an appropriate signature scheme Σ with parameters $\Sigma.\mathsf{pk}$, $\Sigma.\mathsf{sk}$, $\Sigma.\mathsf{sign}$ and $\Sigma.\mathsf{verify}$.
- **ConfirmSign queries.** For a signature query on a message m. $\mathcal{R}$ first computes a (digital) signature σ on m using his secret key $\Sigma.\mathsf{sk}$. Then, he encrypts σ and outputs the result to $\mathcal{A}$. Besides, $\mathcal{R}$ issues a ZK proof of knowledge of σ that satisfies the equation defined by $\Sigma.\mathsf{verify}$. Such a proof is possible for $\mathcal{R}$ to provide since he knows the randomness used to encrypt σ and the relation between

$\Gamma.\mathsf{encrypt}(\sigma)$ and σ defines an NP language and thus accepts a zero knowledge proof system according to [21]. Finally, $\mathcal{R}$ will maintain a list $\mathcal{L}$ of the queries (messages), the corresponding digital signatures and finally the signatures he issued.

Selective conversion queries For a putative confirmer signature μ on m, $\mathcal{R}$ will look up the list $\mathcal{L}$. We note that each record of $\mathcal{L}$ comprises three components : (1) the queried message m_i (2) σ_i corresponding to a digital signature on m_i (3) $\Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(\sigma_i) = \mu_i$, which corresponds to the confirmer signature issued on m_i. If no record having as first component the message m appears in $\mathcal{L}$, then $\mathcal{R}$ will output $\perp$. Otherwise, let t be the number of records having as first component the message m. $\mathcal{R}$ will invoke the plaintext checking oracle (PCA) furnished by his own challenger on (σ_i, μ), for $1 \leq i \leq t$, where σ_i corresponds to the second component of such records. If the PCA oracle identifies μ as a valid encryption of some σ_i, $1 \leq i \leq t$, then $\mathcal{R}$ will return σ_i, otherwise he will return $\perp$. This simulation differs from the real one when the signature μ is valid and was not obtained from the signing oracle. Since the only ways to create a valid confirmer signature which was not issued by $\mathcal{R}$ is either to encrypt a digital signature obtained from the conversion oracle or to come up with a new fresh pair of message and corresponding signature (m, μ). $\mathcal{R}$ can handle the first case using his PCA oracle and list of records $\mathcal{L}$. In the second case, we can distinguish two subcases: either m has not been queried to the signing oracle in which case the pair (m, μ) corresponds to an existential forgery on the confirmer signature scheme and thus to an existential forgery on the underlying digital scheme according to Theorem 1, or m has been queried to the signing oracle but $\Gamma.\mathsf{decrypt}(\mu)$ is not an output of the selective conversion oracle, which corresponds to a strong existential forgery on the underlying digital signature. Therefore, the probability that this scenario does not happen is at least $(1 - \epsilon')^{q_{sc}}$ because the underlying digital signature scheme is (t, ϵ', q_s)-SEUF-CMA secure by assumption.

Verification (Confirmation/denial) queries $\mathcal{R}$ will proceed exactly as in the selective conversion with the exception of simulating the denial protocol instead of returning $\perp$, or the confirmation protocol instead of returning the converted digital signature. $\mathcal{R}$ can issue such proofs without knowing the private key of the cryptosystem using the rewinding technique (See [33] for an illustration) because the protocols are zero knowledge and thus simulatable, or using designated verifier proofs [27] in a registrated key model. Analogously, the probability that $\mathcal{A}$ does not query a valid signature he has not obtained from the signing oracle is at least $(1 - \epsilon')^{q_v}$.

Challenge. Eventually, $\mathcal{A}$ he will output two challenging messages m_0 and m_1. $\mathcal{R}$ will then compute two signatures σ_0 and σ_1 on m_0 and m_1 respectively, which he gives to his own challenger. $\mathcal{R}$ will receive then the challenge $\mu^\star$, as the encryption of either σ_0 or σ_1, which he will forward to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ will continue issuing queries to the signing, confirmation/denial and selective conversion oracles and $\mathcal{R}$ can answer as previously. Note that in this phase, $\mathcal{A}$ is not allowed to query the signing oracle for a signature on m_0, m_1 or the selective conversion, confirmation/denial oracle on $(m_i, \mu^\star)$, $i = 0, 1$. Also, $\mathcal{R}$ is not allowed to query his PCA oracle on $(\mu^\star, \sigma_i)$, $i = 0, 1$. If during the selective conversion or confirmation/denial queries made by $\mathcal{A}$, $\mathcal{R}$ is compelled to query his PCA oracle on $(\mu^\star, \sigma_i)$, $i = 0, 1$, he will simply output $\perp$ in case of a selective conversion query or simulate the denial protocol in case of a verification query. This differs from the real scenario when $\mu^\star$ is a valid confirmer signature on some message $m \notin \{m_0, m_1\}$, which corresponds to an existential forgery on the underlying signature scheme. Again, this happens with probability at most $\epsilon'^{q_{sc}+q_v}$.

Final output. When $\mathcal{A}$ outputs his answer $b \in \{0, 1\}$, $\mathcal{R}$ will forward this answer to his own challenger. Therefore $\mathcal{R}$ will IND-PCA break the underlying cryptosystem with advantage at least $\epsilon \cdot (1 - \epsilon')^{(q_v + q_{sc})}$, in time at most $t + q_s q_{sc}(q_v + q_{sc})$ after at most $q_{sc}(q_{sc} + q_v)$ queries to the PCA oracle. □

E Efficient Instantiations using Certain Signatures and Cryptosystems

E.1 Proof of Theorem 7

We first remark that the function f used in the definition of the class $\mathbb{S}$ induces a group law in $\mathbb{H} = f(\mathbb{G})$ for the operation $\circ_s$. Moreover, we have $1_{\mathbb{H}} = f(1_{\mathbb{G}})$ and $\forall S \in \mathbb{G}$: $f(s)^{-1} = f(s^{-1})$.

Proof. For completeness, it is clear that if both parties follow the protocol, the prover will always be able to provide a proof that the verifier will accept.
For soundness, we show that the prover can cheat with a probability at most 2^{-1} in one round. In fact, suppose that the prover can answer both challenges for the same commitment t_1. Let s_0 and s_1 be the responses of the prover to the challenges 0 and 1 respectively in Step 3. Since the verifier accepts the proof, we have, $t_1 = f(s_0) \circ_s I = f(s_1)$. Thus, $f(s_1) \circ_s f(s_0)^{-1} = f(s_1 * s_0^{-1}) = I$. Hence, the prover would know a preimage of I. We conclude that a cheating prover can cheat with at most $1/2$, provided f is one-way and the verifier is honest (chooses the bit c uniformly from $\{0, 1\}$). Repeating the protocol l times leads to a soundness error which is at most 2^{-l}.

To prove that the proof is ZK, we provide the following simulator.

1. Generate uniformly a random bit $c' \in_R \{0, 1\}$. If $c' = 0$, choose $s' \in_R \mathbb{G}$ and sends $t_1 = f(s') * I$, otherwise, choose $s'' \in_R \mathbb{G}$ and sends $t_1 = f(s'')$ to the verifier.
2. Get c from the verifier. If $c = c'$: if $c = 0$, the simulator sends back s', otherwise, it sends s''. If $c \neq c'$, it goes to Step 1.

The prover's first message is always the function f applied to a random value $s'' \in \mathbb{G}$, and so is the first message of the simulator. Since c' is chosen uniformly at random from $\{0, 1\}$, the probability that the simulator rewinds the verifier is:

$$1 - \Pr[c = c'] = 1 - (\Pr[c = 0, c' = 0] + \Pr[c = 1, c' = 1]) = 1 - (\frac{1}{2}p + \frac{1}{2}(1 - p)) = 1 - \frac{1}{2} = \frac{1}{2}$$

where $p = \Pr[c = 0]$. Therefore, the expected number of rewinds is 2 and as a consequence, the simulator runs in expected linear time. Finally, the distribution of the answers of the prover and of the simulator is again the same. We conclude that the protocol is ZK. It also remains ZK if it is run l times in parallel, where l is either constant or logarithmic in the security parameter. In fact, the simulator of the parallel composition of the protocol will be the parallel composition of the above simulator. Thus, the expected running time of the new simulator is 2^l (probability of not rewinding the verifier is 2^{-l}), which is either constant or polynomial in the security parameter. □

E.2 A variant of the Paillier encryption in the KEM/DEM framework

The Paillier encryption [35] operates on messages in $\mathbb{Z}_N$, where N is a safe RSA modulus. Encryption of a message m is done by picking a random $r \in_R \mathbb{Z}_N^\times$ and then computing the ciphertext $c = r^N(1 + mN) \bmod N^2$. We propose the following KEM/DEM-based variant. To encrypt a message $m \in \mathbb{Z}_N$, first pick a random $k \in_R \mathbb{Z}_N$, encrypt it in c using Paillier's encryption and then output $(c, k + m)$ as the encryption of m. Decryption is done by first "decapsulating" c to recover k, and then subtracting k from the second component of the ciphertext.

The new cryptosystem is in $\mathbb{E}$ since the product $\circ_e$ of two arbitrary ciphertexts is the ciphertext corresponding to the sum of their underlying plaintexts. We define the product $\circ_e$ in $\mathbb{Z}_{N^2}^\times \times \mathbb{Z}_N$ to be $(a, b) \circ_e (c, d) = (ac \bmod N^2, b + d \bmod N)$. Moreover, if r_0 and r_1 are the randomness used in two arbitrary ciphertexts c_0 and c_1, then $r_0 r_1$ is the randomness used to encrypt the sum of the plaintexts corresponding to c_0 and c_1 resp in $c_0 \circ_e c_1$.

Moreover, the new cryptosystem is IND-CPA secure provided the original one is also IND-CPA secure. In fact, let $\mathcal{A}$ be an IND-CPA adversary against the new cryptosystem. We build and IND-CPA adversary $\mathcal{R}$ against the original cryptosystem as follows. When $\mathcal{A}$ outputs his two challenge messages m_0 and m_1. $\mathcal{R}$ will choose a random $k_0 \in_R \mathbb{Z}_N$, then compute $k_1 = m_0 + k_0 - m_1$ and finally output k_0 and k_1 to his challenger. $\mathcal{R}$ will get $c^\star$, as the encryption of either k_0 or k_1, that he will forward to $\mathcal{A}$ along with $k_0 + m_0$ (equal to $k_1 + m_1$ by construction). The output of $\mathcal{A}$ is sufficient for $\mathcal{R}$ to conclude.

We discuss now the security of the underlying KEM and DEM. Concerning the KEM, we need to prove that it is IND-CPA secure. i.e., prove that given some encapsulation c, it is difficult to distinguish $\mathsf{decap}(c)$ from a random element in the corresponding space. Note that the corresponding space is given by the security parameter and not a specific modulus N. Moreover, we know that given an encapsulation c, which corresponds to an encryption of some, element $k \in \mathbb{Z}_N$ using Paillier's encryption, is hard to distinguish k from a random element in $\mathbb{Z}_N$ if the Decisional Composite Residuosity Assumption (corresponding to distinguishing N-th residues from random elements in $\mathbb{Z}_{N^2}$ see [35]) holds. Therefore, one way to extend this indistinguishability to elements in the ciphertext space (given only by the security parameter $|N|$, i.e., the bit length of the modulus N), one can use the recent technique, used in the area of undeniable signatures by [34] (Section 6.1), which consists in using "close enough" moduli, i.e., moduli with common high leading bits, say about 80 (It is known how to generate moduli N with about $|N|/2$ leading bit [1]). In this way, the ciphertext space is about the same (in the view of a polynomial attacker) for all the considered moduli and therefore indistinguishability of some element w.r.t. some modulus will induce indistinguishability for w.r.t. the other moduli. Finally the DEM used in the above cryptosystem is obviously INV-OT secure. In fact, by construction k is random and since the DEM encryption function is one-to-one, then so is the result $m + k$. Therefore, ciphertexts obtained from the DEM are statistically indistinguishable from random elements in the ciphertext space, namely $\mathbb{Z}_N$. We extend this indistinguishability to the ciphertext space by the same argument of the "close enough" moduli.

E.3 Proof of Theorem 9

Proof. The confirmation protocol depicted in Figure 3 is a parallel composition of the proofs depicted in Figures 1 and 2. Therefore completeness and soundness follow as a direct consequence of the completeness and soundness of the underlying proofs (see [20]).

To prove that the protocol is ZK. We provide the following simulator (for one execution):

1. Generate $c' \in_R \{0,1\}$. If $c' = 0$, choose $s' \in_R \mathbb{G}$ and sends $t_1 = f(s') * I$ and $t_2 = \mathsf{encrypt}(s') * (e, s_k)$, otherwise, choose $s'' \in_R \mathbb{G}$ and sends $t_1 = f(s'')$ and $t_2 = \mathsf{encrypt}(s'')$ to the verifier.
2. Get c from the verifier. If $c = c'$: if $c = 0$, the simulator sends back s' and the randomness used to encrypt $\mathsf{encrypt}(s')$, otherwise, it sends s'' and simulates the proof of t_2 being an encryption of s'' (this proof is simulatable since it is by assumption ZK). If $c \neq c'$, it goes to Step 1.

The prover's first message is an encryption of a random value $s'' \in_R \mathbb{G}$, in addition to $f(s'')$, and so is the simulator's first message. Therefore the distributions of the prover and of the simulator are the same in the first round of the proof. Moreover, the expected number of rewinds is two ($\Pr(c \neq c') = \frac{1}{2}$), making the simulator run in an expected linear time. The distribution of the prover's messages in the third round are also similar to those of the simulator. We conclude that the confirmation protocol is ZK. Parallel execution of the protocol will remain also ZK if the number of executions l is constant or logarithmic in the security parameter (see the above proof). □

E.4 Proof of Theorem 10

Proof. With the standard techniques, we prove that the denial protocol depicted in Figure 3 is complete and sound with error probability 2^{-l} (l is the number of rounds) provided the verifier is honest and the cryptosystem is one way. Similarly, we provide the following simulator to prove the ZK property.

1. Generate $c' \in_R \{0,1\}$. If $c' = 0$, choose $s' \in_R \mathbb{G}$ and sends $t_1 = f(s') * I$ and $t_2 = \mathit{\Gamma}.\mathsf{encrypt}(s') * (e, s_k)$, otherwise, choose $s'' \in_R \mathbb{G}$ and a random $t_1 \in_R f(\mathbb{G})$ and $t_2 = \mathit{\Gamma}.\mathsf{encrypt}(s'')$.
2. Get c from the verifier. If $c = c'$: if $c = 0$, the simulator sends back s' and the randomness used to encrypt $\mathit{\Gamma}.\mathsf{encrypt}(s')$, otherwise, it sends s'' and simulates the proof of t_2 being an encryption of s'' (this proof is simulatable since it is by assumption zero knowledge). If $c \neq c'$, it goes to Step 1.

The prover's first message is an encryption of some random value s'' and the element $t_1 = f(s'' * s^{-1}) \circ_s I$. The simulator's first message is an encryption of a random value s'', and in case $b = 0$ the element $t_1 = f(s'' * s^{-1}) \circ_s I$, whereas in the case $b = 1$, it is the element $t_1 \in_R f(\mathbb{G})$ (independent of s''). Distinguishing these two cases it at least as hard as breaking the IND-CPA security of the underlying cryptosystem. In fact, if the verifier is able to distinguish these two cases, it can be easily used to break the cryptosystem in the IND-CPA sense. Therefore, under the assumption of the IND-CPA security of the cryptosystem, the simulator's and prover's first message distributions are indistinguishable. Moreover, the simulator runs in expected linear time, since the number of rewinds is 2. Moreover, the distribution of the prover's and the simulator's message in the last round are again, by the same argument, indistinguishable under the IND-CPA security of the cryptosystem. Finally, with same argument as above, parallel execution of the protocol remains also ZK if the number of executions is constant or logarithmic in the security parameter. $\square$