Generic Constructions for Verifiable Signcryption

Laila El Aimani

Technicolor, 1 avenue de Belle Fontaine - CS17616, 35576 Cesson-Sévigné, France

Abstract. Signcryption is a primitive which simultaneously performs the functions of both signature and encryption in a way that is more efficient than signing and encrypting separately. We study in this paper constructions of signcryption schemes from basic cryptographic mechanisms; our study concludes that the known constructions require expensive encryption in order to attain confidentiality, however some adjustments make them rest on cheap encryption without compromising their security. Our constructions further enjoy verifiability which entitles the sender or the receiver to prove the validity of a signcryption with/out revealing the *signcrypted* message. They also allow the receiver to release some information which allows anyone to publicly verify a signcryption on a given message. Finally, our constructions accept efficient instantiations if the building blocks belong to a wide class of signature/encryption schemes.
Keywords: signcryption, sign-then-encrypt paradigm, commit-then-encrypt-and sign paradigm, encrypt-then-sign paradigm, (public) verifiability, homomorphic encryption.

1 Introduction

Cryptographic mechanisms that proffer both the functionalities of signature and of encryption are becoming nowadays increasingly important. In fact, many real-life applications entail both the confidentiality and the authenticity/integrity of the transmitted data; an illustrative example is electronic elections where the voter wants to encrypt his vote to guarantee privacy, and at the same time, the voting center needs to ensure that the encrypted vote comes from the entity that claims to be its provenance. To respond to this need, Zheng [51] introduced the notion of *signcryption* which is a primitive that simultaneously performs the functions of both signature and encryption in a way that is more efficient than signing and encrypting separately.

Related work Since the introduction of this primitive, many constructions which achieve different levels of security have been proposed. On a high level, security of a signcryption scheme involves two properties; privacy and authenticity. Privacy is analogous to indistinguishability in encryption schemes, and it denotes the infeasibility to infer any information about the *signcrypted* message. Authenticity is similar to unforgeability in signature schemes and it denotes the difficulty to impersonate the *signcrypter*. Defining formally those two properties is a fundamental divergence in signcryption constructions as there are many issues which come into play:

– TWO-USER VERSUS MULTI-USER SETTING In the two-user setting, adopted for instance in [1], a single sender (the entity that creates the signcryption) interacts with a single receiver (the entity that recovers the message from the signcryption). Although such a setting is too simplistic to represent the reality, e.g. the case of electronic elections, it provides however an important preliminary step towards modeling and building schemes in the multi-user setting. In fact, many works have proposed simple tweaks in order to derive multi-user security from two-user security [1, 39].
– INSIDER VERSUS OUTSIDER SECURITY Another consequential difference between security models is whether the adversary is external or internal to the entities of the system. The former case corresponds to outsider security, e.g. [22], whereas the latter denotes insider security which protects the system protagonists even when some of their fellows are malicious or have compromised/lost their private keys [1, 39]. It is naturally possible to mix these notions into one single signcryption scheme, i.e. insider indistinguishability and outsider unforgeability [1, 16], or outsider indistinguishability and insider unforgeability [2]. However, the most frequent mix is the latter as illustrated by the number of works in the literature, e.g. [1, 34, 2]; it is also justified by the necessity to protect the sender from anyone trying to impersonate him including entities in the system. Insider indistinguishability is by contrast needed in very limited applications; the typical example [1] is when the adversary happens to steal the private key of the sender, thus when it is able to send "fake" messages, but we still wish to protect the privacy of the recorded signcryptions sent by the genuine sender.

– VERIFIABILITY A further requirement on signcryption is verifiability which consists in the possibility to prove efficiently the validity of a given signcryption, or to prove that a signcryption has indeed been produced on a given message. In fact, if we consider the example of electronic elections, the voting center might require from the voter a proof of validity of the ”signcrypted” vote. Also, the trusted party (the receiver) that decrypts the vote might be compelled, for instance to resolve some later disputes, to prove that the sender has indeed produced the vote in question; therefore, it would be desirable to support the prover with efficient means to provide such a proof without having to disclose his private input. This property is also needed in filtering out spams in a secure email system. Although a number of constructions [3, 46, 17, 38, 45] have tackled the notion of verifiability (this notion is often referred to in the literature as public verifiability, and it denotes the possibility to release (by the receiver) some information which allows to publicly verify a signcryption with/out revealing the message in question), most of these schemes do not allow the sender to prove the validity of the created signcryption, nor allow the receiver to prove *without revealing any information, ensuring consequently non-transferability*, to a third party, the validity of a signcryption w.r.t. a given message. It is worth noting that the former need, i.e. allowing the sender to prove the validity of a signcryption without revealing the message, already manifests in the IACR electronic voting scheme (The Helios voting scheme) where the sender proves the validity of the encrypted vote to the voting manager. The scheme nonetheless does not respond to the formal security requirements of a signcryption scheme.

Before ending this paragraph, we recall the main generic constructions of signcryption schemes that were proposed so far. In fact, building complex mechanisms from basic ones is customary in cryptography as it allows achieving easy-to-analyze schemes, compared to dedicated/monolithic constructions. The first constructions of signcryption were given and analyzed in [1], where the authors study how to derive signcryption schemes, mainly in the two-user setting, using the classical combinations “sign-then-encrypt”, “encrypt-then-sign”, and “commit-then-encrypt-and-sign”. Subsequently, the work [39] presented several optimizations of these combinations that lead to signcryptions with multi-user security. The paper shows also how to use symmetric encryption in order to derive constructions in the outsider multi-user setting. Finally, there are the recent constructions [16] which achieve security in the insider multi-user setting without key registration assumptions (on the receiver’s side). It is worth noting that none of these constructions treat verifiability.

To the best of our knowledge, there are no generic constructions which provide verifiability in a reasonable security model. The main contribution of this paper is to provide such constructions.

Our Contributions We make the following contributions. First, we propose a new model for signcryption schemes which upgrades the existing models by three interactive protocols: 1. a protocol that allows the sender to prove, to a third party, the validity of the created signcryption, 2. and two protocols that allow the receiver to prove, to a third party, the validity of a given signcryption with/out revealing the message. All these protocols do not require the provers to reveal *any information*.

In Section 3, we show that the “sign-then-encrypt” (StE) and the “commit-then-encrypt-and-sign” (CtEaS) paradigms require expensive assumptions on the underlying encryption in order to derive signcryption with outsider indistinguishability. We do this by first proving the insufficiency of OW-CCA and NM-CPA secure encryption, then by exhibiting a simple attack if the system is instantiated from certain encryption schemes. Next, we propose simple tweaks of the paradigms that make the resulting constructions rest on cheap encryption.

In Section 4, we show that the “encrypt-then-sign” (“TagEncrypt-then-sign”) paradigm provides efficient constructions which are proven secure in our adopted model. We demonstrate the efficiency of these schemes by explicitly describing the different verification protocols if the constructions are instantiated from a wide class of encryption (tag-based encryption) schemes.

Finally, in Section 5, we propose a new paradigm which combines the merits of both the “sign-then-encrypt” (StE) and “encrypt-then-sign” (EtS) paradigms while avoiding their drawbacks. In fact, the former (both the old and the new variant) suffers the problem of verifiability as the underlying encryption operates on bit-strings rather than elements from algebraic sets where homomorphisms could be used in order to ease verifiability. The latter suffers the recourse to stronger security assumptions on the underlying signature in order to get outsider indistinguishability. Moreover, the paradigm does not provide anonymity of the sender. In this section, we show that our new proposed paradigm, called “encrypt-then-sign-then-encrypt” (EtStE) circumvents these problems while accepting many efficient instantiations.

2 Model and Main Constructions

In this section, we present our model for verifiable signcryption. We refer to Appendix A for the necessary cryptographic bricks that will come into play, namely digital signatures, public key encryption schemes, tag-based encryption, KEM/DEM mechanisms, and commitment schemes.

A verifiable signcryption scheme consists of the following algorithms/protocols:

Setup ($\mathsf{setup}(1^\kappa)$). This probabilistic algorithm inputs a security parameter κ, and generates the public parameters param of the signcryption scheme.

Key generation ($\mathsf{keygen}_U(1^\kappa, \mathsf{param}), U \in \{S, R\}$). This probabilistic algorithm inputs the security parameter κ and the public parameters param, and outputs a key pair $(\mathsf{pk}_U, \mathsf{sk}_U)$ for the system user U which is either the sender S or the receiver R.

Signcryption ($\mathsf{signcrypt}(m, \mathsf{sk}_S, \mathsf{pk}_S, \mathsf{pk}_R)$). This probabilistic algorithm inputs a message m, the key pair $(\mathsf{sk}_S, \mathsf{pk}_S)$ of the sender, the public key pk_R of the receiver, and outputs the signcryption μ of the message m.

Proof of validity ($\mathsf{proveValidity}(\mu, \mathsf{pk}_S, \mathsf{pk}_R)$). This is an interactive protocol between the receiver or the sender who has just generated a signcryption μ on some message, and any verifier: the sender uses the randomness used to create μ (as private input) and the receiver uses his private key sk_R in order to convince the verifier that μ is a valid signcryption on some message. The common input to both the prover and the verifier comprise the signcryption μ in question, pk_S, and pk_R. At the end of the protocol, the verifier either accepts or rejects the proof.

Unsigncryption ($\mathsf{unsigncrypt}(\mu, \mathsf{sk}_R, \mathsf{pk}_R, \mathsf{pk}_S)$). This is a deterministic algorithm which inputs a putative signcryption μ on some message, the key pair $(\mathsf{sk}_R, \mathsf{pk}_R)$ of the receiver, and the public key pk_S of the sender, and outputs either the message underlying μ or an error symbol $\perp$.

Confirmation/Denial ($\{\mathsf{confirm}, \mathsf{deny}\}(\mu, m, \mathsf{pk}_R, \mathsf{pk}_S)$). These are interactive protocols between the receiver and any verifier; the receiver uses his private key sk_R (as private input) to convince any verifier that a signcryption μ on some message m is/is not valid. The common input comprises the signcryption μ and the message m in question, in addition to pk_R and pk_S. At the end of the protocol, the verifier is either convinced of the validity/invalidity of μ w.r.t. m or not.

Public verification ($\mathsf{publicVerify}(\mu, m, \mathsf{sk}_R, \mathsf{pk}_R, \mathsf{pk}_S)$). This is an algorithm which inputs a signcryption μ, a message m, the key pair $(\mathsf{sk}_R, \mathsf{pk}_R)$ of the receiver, and the public key pk_S of the sender, and outputs either an error symbol $\perp$ if μ is not a valid signcryption on m, or a string which allows to publicly verify the validity of μ on m otherwise.

Remark 1.
- The $\mathsf{proveValidity}$ protocol allows the sender to prove the validity of the signcryption he has just created (need for the randomness used to produce the signcryption). Although this situation is plausible in secure email or in electronic elections, it would be however nice to have a stateless system. This would require in our constructions involved non-interactive proofs which are in general difficult to obtain in the standard model.
- The $\{\mathsf{confirm}, \mathsf{deny}\}$ protocols can also be run by the sender who has just generated the signcryption in question. Furthermore, they are interactive in order to ensure *non-transferability*, i.e. the possibility of the verifier to transfer to a third party his conviction about the validity/invalidity of a signcryption w.r.t. a given message. It has been proven in [37] that interactivity guarantees only *offline non-transferability*, i.e., non-transferability is not preserved if the verifier interacts concurrently with the receiver and an unexpected verifier. One way to remediate to this problem was proposed in [18] using non-interactive designated verifier proofs. Again, the proposed solution rests on heavy non-interactive proofs using the [30] proof system, and is hard to be generalized.

It is natural to require the correctness of a signcryption scheme, i.e. for any message m:

$$\mathsf{unsigncrypt}(\mathsf{signcrypt}(m, \mathsf{sk}_S, \mathsf{pk}_S, \mathsf{pk}_R), \mathsf{sk}_R, \mathsf{pk}_R, \mathsf{pk}_S) = m.$$

and

$$\mathsf{publicVerify}(m, \mathsf{signcrypt}(m, \mathsf{sk}_S, \mathsf{pk}_S, \mathsf{pk}_R), \mathsf{sk}_R, \mathsf{pk}_R, \mathsf{pk}_S) \neq \perp .$$

Moreover, the protocols $\mathsf{proveValidity}$ and $\{\mathsf{confirm}, \mathsf{deny}\}$ must be complete, sound, and zero knowledge. We refer to [27] for details of these notions.

2.1 Unforgeability

This notion protects the sender's authenticity from *malicious insider* adversaries, i.e. the receiver. It is defined through a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ where the latter gets the public key pk_S of the sender, generates the key pair $(\mathsf{pk}_R, \mathsf{sk}_R)$ of the receiver, and hands pk_R to the challenger. During the game, $\mathcal{A}$ is allowed to ask adaptively signcryption queries w.r.t. pk_R and pk_S on messages of his choice to $\mathcal{C}$. The scheme is said to be *Existentially Unforgeable against Chosen Message Attacks (EUF-CMA)* if the adversary is unable to produce a valid signcryption $\mu^\star$ on a message $m^\star$ that he did not ask to the signcryption oracle.

Definition 1 (Unforgeability). *We consider a signcryption scheme* sc *given by the algorithms/protocols defined earlier in this section. Let $\mathcal{A}$ be a PPTM. We consider the following random experiment:*

$\boxed{\textit{Experiment } \mathbf{Exp}^{\mathsf{euf\text{-}cma}}_{\mathsf{sc},\mathcal{A}}(1^\kappa)}$

$\mathsf{param} \leftarrow \mathsf{sc.setup}(1^\kappa)$
$(\mathsf{pk}_S, \mathsf{sk}_S) \leftarrow \mathsf{sc.keygen}_S(1^\kappa, \mathsf{param})$
$\mathsf{pk}_R \leftarrow \mathcal{A}(\mathsf{pk}_S)$
$\mu^\star \leftarrow \mathcal{A}^{\mathfrak{S}}(\mathsf{pk}_S, \mathsf{pk}_R, \mathsf{sk}_R)$
$\quad\mathfrak{S} : m \longmapsto \mathsf{sc.signcrypt}\{\mathsf{sk}_S, \mathsf{pk}_S, \mathsf{pk}_R\}(m)$
return 1 if and only if the following properties are satisfied:
- $\mathsf{sc.unsigncrypt}_{\{\mathsf{sk}_R, \mathsf{pk}_R, \mathsf{pk}_S\}}[\mu^\star] = m^\star$
- $m^\star$ *was not queried to* $\mathfrak{S}$

We define the success *of* $\mathcal{A}$ via*:*

$$\mathbf{Succ}^{\textit{euf-cma}}_{\mathsf{sc},\mathcal{A}}(1^\kappa) = \Pr\left[\mathbf{Exp}^{\textit{euf-cma}}_{\mathsf{sc},\mathcal{A}}(1^\kappa) = 1\right].$$

Given $(t, q_s) \in \mathbb{N}^2$ and $\varepsilon \in [0, 1]$, $\mathcal{A}$ is called a (t, ε, q_s)-EUF-CMA adversary against sc *if, running in time t and issuing q_s queries to the* sc.signcrypt *oracle, $\mathcal{A}$ has $\mathbf{Succ}^{\textit{euf-cma}}_{\mathsf{sc},\mathcal{A}}(1^\kappa) \geq \varepsilon$. The scheme* sc *is said to be (t, ε, q_s)-EUF-CMA secure if no (t, ε, q_s)-EUF-CMA adversary against it exists.*

Remark 2. Note that $\mathcal{A}$ in the above definition is not given the oracles sc.proveValidity, sc.unsigncrypt, sc.publicVerify, and sc.$\{$confirm, deny$\}$. In fact, these oracles are useless for him as he has the receiver's private key sk_R at his disposal.

2.2 Indistinguishability

This notion protects the sender's privacy from *outsider adversaries*. It is defined through a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$; $\mathcal{C}$ generates the key pairs $(\mathsf{sk}_S, \mathsf{pk}_S)$ and $(\mathsf{sk}_R, \mathsf{pk}_R)$ for the sender and for the receiver respectively, and hands $(\mathsf{pk}_S, \mathsf{pk}_R)$ to $\mathcal{A}$. During the first phase of the game, $\mathcal{A}$ queries adaptively signcrypt and proveValidity (actually proveValidity is only invoked on inputs just obtained from the signcryption oracle), unsigncrypt, $\{$confirm, deny$\}$, and publicVerify for any input. Once $\mathcal{A}$ decides that this phase is over, he generates two messages $m_0^\star, m_1^\star$ and hands them to $\mathcal{C}$ who generates a signcryption $\mu^\star$ on $m_b^\star$ for $b \xleftarrow{R} \{0, 1\}$ and gives it $(\mu^\star)$ to $\mathcal{A}$. The latter resumes querying the previous oracles adaptively on any input with the exception of not querying unsigncrypt on $\mu^\star$, and $\{$confirm, deny$\}$ and publicVerify on the pair $(\mu^\star, m_i^\star)$ for $i \in \{0, 1\}$. At the end, the adversary outputs his guess b' for the message underlying the signcryption $\mu^\star$. He is considered successful if $b = b'$.

Definition 2 (Indistinguishability (IND-CCA)). *Let* sc *be a signcryption scheme, and let $\mathcal{A}$ be a PPTM. We consider the following random experiment for $b \xleftarrow{R} \{0, 1\}$:*

$\boxed{\textit{Experiment } \mathbf{Exp}^{\textit{ind-cca-b}}_{\mathsf{sc},\mathcal{A}}(1^\kappa)}$

$\mathsf{param} \leftarrow \mathsf{sc.setup}(1^\kappa)$
$(\mathsf{sk}_S, \mathsf{pk}_S) \leftarrow \mathsf{sc.keygen}_S(1^\kappa, \mathsf{param})$
$(\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow \mathsf{sc.keygen}(1^\kappa, \mathsf{param})$
$(m_0^\star, m_1^\star, \mathcal{I}) \leftarrow \mathcal{A}^{\mathfrak{S},\mathfrak{V},\mathfrak{U},\mathfrak{C}}(\textit{find}, \mathsf{pk}_S, \mathsf{pk}_R)$

$\mathfrak{S} : m \longmapsto \mathsf{sc.signcrypt}_{\{\mathsf{sk}_S,\mathsf{pk}_S,\mathsf{pk}_R\}}(m)$
$\mathfrak{V} : \mu \longmapsto \mathsf{sc.proveValidity}(\mu, \mathsf{pk}_S, \mathsf{pk}_R)$
$\mathfrak{U} : \mu \longmapsto \mathsf{sc.unsigncrypt}_{\mathsf{sk}_R,\mathsf{pk}_R,\mathsf{pk}_S}(\mu)$
$\mathfrak{C} : (\mu, m) \longmapsto \mathsf{sc.}\{\mathsf{confirm}, \mathsf{deny}\}(\mu, m, \mathsf{pk}_R, \mathsf{pk}_S)$
$\mathfrak{P} : (\mu, m) \longmapsto \mathsf{sc.publicVerify}(\mu, m, \mathsf{pk}_R, \mathsf{pk}_S)$

$\mu^\star \leftarrow \mathsf{sc.signcrypt}_{\{\mathsf{sk}_S,\mathsf{pk}_S,\mathsf{pk}_R\}}(m_b^\star)$
$d \leftarrow \mathcal{A}^{\mathfrak{S},\mathfrak{V},\mathfrak{U},\mathfrak{C}}(\textit{guess}, \mathcal{I}, \mu^\star, \mathsf{pk}_S, \mathsf{pk}_C)$

$\mathfrak{S} : m \longmapsto \mathsf{sc.signcrypt}_{\{\mathsf{sk}_S,\mathsf{pk}_S,\mathsf{pk}_R\}}(m)$
$\mathfrak{V} : \mu \longmapsto \mathsf{sc.proveValidity}(\mu, \mathsf{pk}_S, \mathsf{pk}_R)$
$\mathfrak{U} : \mu(\neq \mu^\star) \longmapsto \mathsf{sc.unsigncrypt}_{\mathsf{sk}_R,\mathsf{pk}_R,\mathsf{pk}_S}(\mu)$
$\mathfrak{C} : (\mu, m)(\neq (\mu^\star, m_i^\star), i = 0, 1) \longmapsto \mathsf{sc.}\{\mathsf{confirm}, \mathsf{deny}\}(\mu, m, \mathsf{pk}_R, \mathsf{pk}_S)$
$\mathfrak{P} : (\mu, m)(\neq (\mu^\star, m_i^\star), i = 0, 1) \longmapsto \mathsf{sc.publicVerify}(\mu, m, \mathsf{pk}_R, \mathsf{pk}_S)$

Return d

We define the advantage *of* $\mathcal{A}$ via*:*

$$\mathbf{Adv}^{\textit{ind-cca}}_{\mathsf{sc},\mathcal{A}}(1^\kappa) = \left|\Pr\left[\mathbf{Exp}^{\mathsf{ind-cca-b}}_{\mathsf{sc},\mathcal{A}}(1^\kappa) = b\right] - \frac{1}{2}\right|.$$

Given $(t, q_s, q_v, q_u, q_{cd}, q_{pv}) \in \mathbb{N}^6$ *and* $\varepsilon \in [0, 1]$*,* $\mathcal{A}$ *is called a* $(t, \varepsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$*-IND-CCA adversary against* sc *if, running in time* t *and issuing* q_s *queries to the* sc.signcrypt *oracle,* q_v *queries to the* sc.proveValidity *oracle,* q_u *queries to the* sc.unsigncrypt *oracle,* q_{cd} *queries to the* sc.{confirm, deny} *oracle, and* q_{pv} *to the* publicVerify *oracle,* $\mathcal{A}$ *has* $\mathbf{Adv}^{\mathsf{ind-cca}}_{\mathsf{sc},\mathcal{A}}(1^\kappa) \geq \varepsilon$*. The scheme* sc *is said to be* $(t, \varepsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$*-IND-CCA secure if no* $(t, \varepsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$*-IND-CCA adversary against it exists.*

In Appendix B, we provide the above properties in the multi-user setting, namely the dM-EUF-CMA and the fM-IND-CCA security properties, where the adversary is further given all the private keys except those of the target sender and of the target receiver.

2.3 Main Constructions

Let Σ be a digital signature scheme given by Σ.keygen which generates a key pair (Σ.sk, Σ.pk), Σ.sign, and Σ.proveValidity. Let furthermore Γ denote a public key encryption scheme described by Γ.keygen that generates the key pair (Γ.sk,Γ.pk), Γ.encrypt, and Γ.decrypt. Finally, let Ω be a commitment scheme given by the algorithms Ω.commit and Ω.open. The most popular paradigms used to devise signcryption schemes from basic primitives are:

- The *"sign-then-encrypt" (StE) paradigm*. Given a message m, signcrypt first produces a signature σ on the message using Σ.sk, then encrypts $m\|\sigma$ under Γ.pk. The result forms the signcryption on m. To unsigncrypt, one first decrypts the signcryption using Γ.sk in $m\|\sigma$, then checks the validity of σ, using Σ.pk, on m. Finally, publicVerify of a valid signcryption $\mu = \Gamma.\mathsf{encrypt}(m\|\sigma)$ on m outputs σ.
- The *"encrypt-then-sign" (EtS) paradigm*. Given a message m, signcrypt produces an encryption e on m using Γ.pk, then produces a signature σ on e using Σ.sk; the signcryption is the pair (e, σ). To unsigncrypt such a signcryption, one first checks the validity of σ w.r.t. e using Σ.pk, then decrypts e using Γ.sk to get m. Finally, publicVerify outputs a zero knowledge non-interactive (NIZK) proof that m is the decryption of e; such a proof is possible since the statement in question is in NP ([28] and [6]).
- The *"commit-then-encrypt-and-sign" (CtEaS) paradigm*. This construction has the advantage of performing the signature and the encryption *in parallel* in contrast to the previous sequential compositions. Given a message m, one first produces a commitment c on it using some random nonce r, then encrypts $m\|r$ under Γ.pk, *and* produces

a signature σ on c using Σ.sk. The signcryption is the triple (e, c, σ). To unsigncrypt such a signcryption, one first checks the validity of σ w.r.t. c, then decrypts e to get $m\|r$, and finally checks the validity of the commitment c w.r.t (m, r). publicVerify is achieved by releasing the decryption of e, namely $m\|r$.

The proofs of well (mal) formed-ness, namely proveValidity and $\{\mathsf{confirm}, \mathsf{deny}\}$ can be carried out since the languages in question are in NP (co-NP) and thus accept zero knowledge proof systems [28]. Finally, it is possible to require a proof in the publicVerify algorithms of StE and CtEaS, that the revealed information is indeed a correct decryption of the encryption in question; such a proof is again possible to issue since the corresponding statement is in NP.

3 Analysis of the StE and CtEaS Paradigms

3.1 Insufficiency of OW-CCA and NM-CPA secure encryption

We proceed in this subsection as in [23] where the author shows the impossibility to derive secure confirmer signatures, using the StE and the CtEaS paradigms, from both OW-CCA and NM-CPA secure encryption; we first show the impossibility result for the so-called *key-preserving reductions*, i.e. reductions which launch the adversary on its challenge public key in addition to some freely chosen parameters, then we generalize the result to arbitrary reductions assuming new assumptions on the underlying encryption scheme.

Lemma 1. *Assume there exists a key-preserving reduction $\mathcal{R}$ that converts an IND-CCA adversary $\mathcal{A}$ against signcryptions from the StE (CtEaS) paradigm to a OW-CCA adversary against the underlying encryption scheme. Then, there exists a meta-reduction $\mathcal{M}$ that OW-CCA breaks the encryption scheme in question.*

This lemma claims that under the OW-CCA security assumption of the underlying encryption, there is no key-preserving reduction that reduces OW-CCA breaking the encryption scheme in question to IND-CCA breaking the signcryption construction (from StE or CtEaS), or if there exists such an algorithm, then the underlying encryption scheme is not OW-CCA secure, thus rendering such a reduction useless.

Proof. Let $\mathcal{R}$ be the key-preserving reduction that reduces OW-CCA breaking the encryption scheme underlying the construction to IND-CCA breaking the construction (from StE or CtEaS) itself. We will construct an algorithm $\mathcal{M}$ that uses $\mathcal{R}$ to OW-CCA break the same encryption scheme by simulating an execution of the IND-CCA adversary $\mathcal{A}$ against the construction.

Let Γ be the encryption scheme $\mathcal{M}$ 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 Γ.pk and the same challenge c. Obviously, all decryption queries made by $\mathcal{R}$, which are by definition different from the challenge c, can be forwarded to $\mathcal{M}$'s own challenger. $\mathcal{M}$ needs now to simulate an IND-CCA adversary $\mathcal{A}$ to $\mathcal{R}$ (Σ and Ω denote respectively the signature and the commitment schemes in use):

- *StE paradigm.* $\mathcal{A}$ receives as a challenge signcryption a certain $\mu_b = \Gamma.\mathsf{encrypt}(\Sigma.\mathsf{sign}(m_b))$, where $b \in \{0, 1\}$ (m_0, m_1 being the challenge messages output by $\mathcal{A}$, simulated by $\mathcal{M}$, to $\mathcal{R}$). With overwhelming probability, $c \neq \mu_b$ (we refer to Remark 3 in case $\mathcal{R}$ misbehaves and submits c as a challenge signcryption) since the challenge c is not encryption of a valid digital signature on the message m_0 or the message m_1. Thus $\mathcal{M}$ can query μ_b to his own challenger for decryption. The answer of such a query is sufficient for $\mathcal{A}$ (simulated by $\mathcal{M}$) to answer his indistinguishability challenge.
- *CtEaS paradigm.* $\mathcal{A}$ receives as a challenge signcryption $\mu_b = [\Gamma.\mathsf{encrypt}(m_b\|r), c = \Omega.\mathsf{commit}(m_b, r), \Sigma.\mathsf{sign}(c)]$ with $b \in \{0, 1\}$ (always m_0, m_1 denote the challenge messages output by $\mathcal{A}$ to $\mathcal{R}$). Similarly, $c \neq \Gamma.\mathsf{encrypt}(m_b\|r)$ (we refer again to Remark 3 in case $\mathcal{R}$ misbehaves and submits c as the first field of the challenge signcryption) with overwhelming probability since c is not encryption of a message whose prefix is the message m_0 or m_1. Thus, $\mathcal{M}$ can query his challenger for the decryption of the first field of μ_b. The result of such a query is sufficient for $\mathcal{A}$ to answer his indistinguishability challenge.

To sum up, $\mathcal{M}$ is able to perfectly answer the decryption queries made by $\mathcal{R}$ (that are by definition different from c). $\mathcal{M}$ is further capable of successfully simulating the IND-CCA challenge against the construction (from the StE or the CtEaS paradigms). Thus $\mathcal{R}$ is expected to return the answer to the OW-CCA challenge, namely the decryption of c. Upon receipt of this answer, $\mathcal{M}$ will forward it to his own challenger. □

Remark 3. In the above proof, if $\mathcal{R}$ submits c in the challenge signcryption to $\mathcal{A}$, then this latter cannot solve the IND-CCA challenge as he ($\mathcal{M}$) cannot invoke his decryption oracle on c. In this case, as it is very unlikely that the resulting signcryption is a valid signcryption on the challenge messages m_0 or m_1, then whatever is the answer of $\mathcal{A}$ (actually in this case, $\mathcal{A}$, simulated by $\mathcal{M}$ who launched $\mathcal{R}$ over c, can abort the indistinguishability game) to the challenge signcryption, this answer will not help $\mathcal{R}$ solving his OW-CCA challenge since he already knows that c cannot be (with overwhelming probability) a valid signcryption on either messages m_0 or m_1. In other words, in this case, whatever $\mathcal{R}$ learns from $\mathcal{A}$, he can also learn it without $\mathcal{A}$, which corresponds to a reduction $\mathcal{R}$ solving a OW-CCA challenge in polynomial time without the help of $\mathcal{A}$, i.e. $\mathcal{R}$ is useless as it is solving an easy problem.

We further have this lemma (which we prove in Appendix C.1) on the insufficiency of NM-CPA secure encryption.

Lemma 2. *Assume there exists a key-preserving reduction $\mathcal{R}$ that converts an IND-CCA adversary $\mathcal{A}$ against signcryptions from the StE (CtEaS) paradigm to a NM-CPA adversary against the underlying encryption scheme. Then, there exists a meta-reduction $\mathcal{M}$ that NM-CPA breaks the encryption scheme in question.*

We generalize in Appendix C.2 the previous results to arbitrary reductions if the encryption scheme has a *non-malleable key generator*, which informally means that OW-CCA (NM-CPA) breaking the encryption, w.r.t. a public key pk, is no easier when given access to a decryption oracle w.r.t. any key pk' different from pk.

Moreover, we can rule out the OW-CPA, OW-PCA, and IND-CPA notions[1] by remarking that ElGamal's encryption meets all those notions (under different assumptions), but cannot be employed in StE and CtEaS as it is malleable. In fact, the indistinguishability adversary can create a new signcryption (by re-encrypting the ElGamal encryption) on the challenge message, and query it for unsigncryption. The answer of such a query is sufficient to conclude. We refer to Appendix C.3 for the detail of this attack using a much larger class of encryption schemes.

In consequence of the above analysis, the used encrypted scheme has to satisfy at least IND-PCA security in order to lead to secure signcryption from StE or CtEaS. Since there are no known encryption schemes in the literature which separate the notions IND-PCA and IND-CCA, our result practically means that the encryption scheme underlying the previous constructions has to satisfy the highest security level (IND-CCA) in order to lead to secure signcryption. This translates in expensive operations, especially if verifiability is further required for the resulting signcryption.

3.2 Positive results

Constructions from StE or CtEaS suffer the strong forgeability: given a signcryption on some message, one can create another valid signcryption on the same message without the sender's help. To circumvent this problem, we propose the following techniques which bind the digital signature to the resulting signcryption.

The new *"sign-then-encrypt"(StE) paradigm*. 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 $\Sigma.\mathsf{proveValidity}$. 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 $\mathcal{D}.\mathsf{decrypt}$. We assume that the message space of $\mathcal{D}$ includes the concatenation of elements from the message space of Σ, and of signatures produced by Σ, and that the encapsulations generated by $\mathcal{K}$ are exactly κ-bit long, where κ is a security parameter.

[1] The step of ruling out OW-CPA, OW-PCA, and IND-CPA is necessary although we have proved the insufficiency of stronger notions, namely OW-CCA and NM-CPA. In fact, suppose there is an efficient key-preserving reduction $\mathcal{R}$ which reduces OW-PCA breaking a cryptosystem Γ underlying a StE or CtEaS construction to IND-CCA breaking the construction itself. Then there exists an efficient key-preserving reduction say $\mathcal{R}'$ that reduces OW-CCA breaking Γ to IND-CCA breaking the construction (since OW-CCA is stronger than OW-PCA). According to the previous Lemmata, such a reduction ($\mathcal{R}'$) may exist if Γ is not OW-CCA secure (although it is OW-PCA secure). In other terms, since there are separations between the notions OW-CCA and OW-PCA (same for the other notions), we cannot apply the insufficiency of OW-CCA (NM-CPA) to rule out the weaker notions.

A signcryption scheme sc is defined as follows: sc.setup invokes the setup algorithms of Σ, $\mathcal{K}$, and $\mathcal{D}$. $\mathsf{sc.keygen}_S$ and $\mathsf{sc.keygen}_R$ consist of Σ.keygen and $\mathcal{K}$.keygen respectively. To sc.signcrypt a message m, one first generates a key k with its encapsulation c using $\mathcal{K}$.encap, then produces a signature σ on $c\|m$, and finally outputs $\mu = (c, \mathcal{D}.\mathsf{encrypt}_k(m\|\sigma))$ as a signcryption of m. Unsigncryption of some (μ_1, μ_2) is done by first recovering the key k from μ_1 using $\mathcal{K}$.decap, then using $\mathcal{D}$.decrypt and k to decrypt μ_2, and finally checking that the result is a valid digital signature on $\mu_1\|m$ where m is the retrieved message. The rest is similar to the original StE.

Theorem 1 *Given $(t, q_s) \in \mathbb{N}^2$ and $\varepsilon \in [0,1]$, 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.*

The proof is given in Appendix C.4.

Theorem 2 *Given $(t, q_s, q_v, q_u, q_{cd}, q_{pv}) \in \mathbb{N}^6$ and $(\varepsilon, \epsilon') \in [0,1]^2$, the above construction is $(t, \epsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$-IND-CCA secure if it uses a (t, ϵ', q_s)-EUF-CMA secure digital signature, an IND-OT secure DEM and an $(t+q_s(q_u+q_{cd}+q_{pv}), \epsilon \cdot (1-\epsilon')^{q_u+q_{cd}+q_{pv}})$-IND-CPA secure KEM.*

The proof is provided in Appendix C.5. We note that the above theorem holds true also when the used DEM is only computationally secure. Details are given in Remark 9.

The new *"commit-then-encrypt-and-sign" (CtEaS) paradigm* The new *"commit-then-encrypt-and-sign" (CtEaS) paradigm*. The construction is similar to the basic one described earlier, with the exception of producing the digital signature on both the commitment c and the encryption e. The new construction looses the parallelism of the original one, i.e. encryption and signature can longer be carried out in parallel, however it has the advantage of resting on cheap encryption compared to the early one.

Theorem 3 *Given $(t, q_s) \in \mathbb{N}^2$ and $(\varepsilon, \epsilon_b) \in [0,1]^2$, the above construction is (t, ϵ, q_s)-EUF-CMA secure if it uses a uses a (t, ϵ_b) binding commitment scheme and a $(t, \epsilon(1-\epsilon_b)^{q_s}, q_s)$-EUF-CMA secure digital signature scheme.*

Theorem 4 *Given $(t, q_s, q_v, q_u, q_{cd}, q_{pv}) \in \mathbb{N}^6$ and $(\varepsilon, \epsilon', \epsilon_h) \in [0,1]^3$, the new CtEaS construction is $(t, \epsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$-IND-CCA secure if it uses a (t, ϵ', q_s)-SEUF-CMA secure digital signature, a statistically binding, and (t, ϵ_h)-hiding commitment, and a $(t + q_s(q_u + q_{cd} + q_{pv}), \frac{1}{2}(\epsilon + \epsilon_h)(1-\epsilon')^{q_u+q_{cd}+q_{pv}})$-IND-CPA secure encryption scheme.*

The proofs are given in Appendix C.6 and Appendix C.7 resp.

4 Efficient Verifiable Signcryption from the EtS Paradigm

The Encrypt-then-Sign paradigm, described in Subsection 2.3, turns out to provide efficient signcryptions schemes that are proven secure in the model we adhere to.

Theorem 5 *Given $(t, q_s) \in \mathbb{N}^2$ and $\varepsilon \in [0,1]$, signcryption schemes from EtS are (t, ϵ, q_s)-EUF-CMA secure if the underlying digital signature scheme is (t, ϵ, q_s)-EUF-CMA secure.*

Theorem 6 *Given $(t, q_s, q_v, q_u, q_{cd}, q_{pv}) \in \mathbb{N}^6$ and $(\varepsilon, \epsilon') \in [0,1]^2$, signcryptions from EtS are $(t, \epsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$-IND-CCA secure if they use a (t, ϵ', q_s)-SEUF-CMA secure digital signature and a $(t + q_s(q_u + q_{cd} + q_{pv}), \epsilon(1-\epsilon')^{q_u+q_{cd}+q_{pv}})$-IND-CPA secure encryption scheme*

We flesh out the details of both proofs in Appendix D.1 and Appendix D.2 respectively. In fact, the proofs that were given so far correspond to security models different from the one we consider.

Remark 4. Note that the IND-CPA requirement on the encryption scheme is also necessary. In fact, an IND-CCA adversary against the signcryption construction can easily use an IND-CPA adversary against the underlying encryption scheme in order to solve his challenge.

4.1 Efficient instantiations

To allow efficient $\mathsf{proveValidity}$, $\{\mathsf{confirm}, \mathsf{deny}\}$, and $\mathsf{publicVerify}$ protocols/algorithms, we propose to instantiate the encryption scheme from the following class:

Definition 3 (The class $\mathbb{E}$ of encryption schemes). *$\mathbb{E}$ is the set of public key encryption schemes Γ that have the following properties:*

1. *The scheme operates in a message space $\mathcal{M}$ which is a group $\mathcal{M} = (\mathbb{G}, *)$, and produces encryptions which belong to a group space $\mathcal{C} = (\mathbb{H}, \circ_e)$, i.e. $\Gamma.\mathsf{encrypt}\colon (\mathbb{G}, *) \to (\mathbb{H}, \circ_e)$.*
2. *Let $m \in \mathcal{M}$ be a message and c its encryption (using $\Gamma.\mathsf{encrypt}$) w.r.t. a key pk. On the common input pk, m, and c, there exists an efficient zero knowledge proof PoK of m being the decryption of c w.r.t. pk. The private input of the prover is either the private key corresponding to pk or the randomness used to produce the encryption c.*
3. *$\forall m, m' \in \mathcal{M}$, $\forall \mathsf{pk}\colon \Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m * m') = \Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m) \circ_e \Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m')$. Moreover, given the randomness used to encrypt m in $\Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m)$ and m' in $\Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m')$, one can deduce (using only the public parameters) the randomness used to encrypt $m * m'$ in $\Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m) \circ_e \Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m')$.*

Examples of encryption schemes in the above class include ElGamal's encryption [24], the encryption scheme defined in [8], or Paillier's [40] encryption scheme. In fact, these schemes are homomorphic and possess efficient protocols for proving that a ciphertext decrypts to a given message: the proof of equality of two discrete logarithms [15], in case of [24, 8], or the proof of knowledge of an N-th root in case of [40].

We describe in the rest of this subsection the $\mathsf{proveValidity}$, $\{\mathsf{confirm}, \mathsf{deny}\}$, and $\mathsf{publicVerify}$ protocols/algorithms if the used encryption belongs to the class $\mathbb{E}$.

Proof of Validity We depict the $\mathsf{proveValidity}$ protocol in Figure 1.

Prover P		Verifier V
Choose $m' \xleftarrow{R} \mathcal{M}$		
Compute $e' = \Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(m')$	$\xrightarrow{\quad e' \quad}$	
	$\xleftarrow{\quad b \quad}$	Choose $b \xleftarrow{R} \{0,1\}^{\ell}$ $(b \in \mathbb{N})$
	$\xrightarrow{\quad z = m' * m^b \quad}$	
	$\xleftrightarrow{\mathsf{PoK}\{e' \circ_e e^b = \Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(z)\}}$	
		Accept if PoK is valid

Fig. 1. Proof system for membership to the language $\{m : e = \Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(m)\}$ Common input: $(e, \Gamma.\mathsf{pk})$ and Private input: m and $\Gamma.\mathsf{sk}$ or randomness used to produce e.

Theorem 7 *Let Γ be a one-way encryption scheme from the class $\mathbb{E}$. The protocol depicted in Figure 1 is a zero knowledge proof of knowledge of the decryption of e.* □

Confirmation/denial protocols The $\mathsf{confirm}$ protocol is nothing but the proof PoK which is in case of [24, 8] a proof of equality of two discrete logarithms, and in case of [40] a proof of knowledge of an N-th root. We depict the deny protocol in Figure 2, where f denotes an arbitrary *homomorphic injective one way function*:

$$\forall m, m'\colon f(m \star m') = f(m) \circ_s f(m')$$

Theorem 8 *Let Γ be an IND-CPA encryption scheme from the above class $\mathbb{E}$. The protocol depicted in Figure 2 is a zero knowledge proof of the decryption of e which is different from the message m.* □

Prover P		Verifier V
Choose $m' \xleftarrow{R} \mathcal{M}$		
Compute $e_1' = f(m')$		
Compute $e_2' = \Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(m')$	$\xrightarrow{\quad e_1', e_2' \quad}$	
	$\xleftarrow{\quad b \quad}$	Choose $b \xleftarrow{R} \{0,1\}^{\ell}$ $(b \in \mathbb{N})$
	$\xrightarrow{\quad z = m' * \tilde{m}^b \quad}$	
	$\overset{\mathsf{PoK}\{e_2' \circ_e e^b = \Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(z)\}}{\longleftrightarrow}$	
		Accept if PoK is valid and if $f(z) \neq e_1' \circ_s f(m)^b$.

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

Public verification the publicVerify algorithm outputs a ZK non-interactive proof of the correctness of a decryption. We note the following three solutions according to the used encryption:

1. The case of Paillier's encryption [40]: this scheme belongs to *fully decryptable* encryption schemes, i.e. encryption schemes where decryption leads to the randomness used to produce the ciphertext. Thus, publicVerify will simply release the randomness used to generate the ciphertext.
2. The case of [8]'s encryption: Groth and Sahai [30] presented an efficient ZK non-interactive proof that a given encryption using this scheme encrypts a given message under a given public key.
3. The case of DL-based encryption schemes, e.g. [24, 8, 20]: the interactive proof of correctness of most such schemes reduces to a proof of equality of two discrete logarithms. The work [21] presented an efficient method to remove interaction using additively homomorphic encryption, e.g. Paillier [40].

4.2 Extension to Multi-user security

The construction is the same provided in [39], namely the TagEncrypt-then-Sign paradigm (TEtS), which deviates from the standard EtS paradigm as follows:

1. It considers a tag-based encryption scheme where the tag is set to the public key of the sender pk_S.
2. The digital signature is produced on the resulting ciphertext and on the public key of the receiver.

Theorem 9 *Given $(t, q_s) \in \mathbb{N}^2$ and $\varepsilon \in [0, 1]$, signcryption schemes from the TEtS paradigm are (t, ϵ, q_s)-dM-EUF-CMA secure if the underlying digital signature scheme is (t, ϵ, q_s)-EUF-CMA secure.*

Theorem 10 *Given $(t, q_s, q_v, q_u, q_{cd}, q_{pv}) \in \mathbb{N}^6$ and $(\varepsilon, \epsilon') \in [0, 1]^2$, signcryption constructions from the TEtS paradigm are $(t, \epsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$-IND-CCA secure if they use a (t, ϵ', q_s)-SEUF-CMA secure digital signature and a $(t+q_s(q_u+q_{cd}+q_{pv}), \epsilon(1-\epsilon')^{q_u+q_{cd}+q_{pv}}, q_u+q_{cd}+q_{pv})$-IND-sTag-CCA secure tag-based encryption scheme.*

We provide again the proofs of both theorems in Appendix D.3 and Appendix D.3 resp. as we consider a model different from that adopted in [39]. Finally, we provide in Appendix D.5 an efficient instantiation of the paradigm using Kiltz'encryption [35].

5 Efficient Verifiable Signcryption from the EtStE Paradigm

The EtS technique compares better with respect to verifiability, since the prover needs simply to prove knowledge of the decryption of a given ciphertext. Also, the receiver has to prove that a message is/isn't the decryption of a given ciphertext. Such proofs are easy to carry out if one considers the already mentioned class $\mathbb{E}$. Moreover, we showed that publicVerify can be made efficient for many encryption schemes from the class $\mathbb{E}$. However, in order to achieve indistinguishability, EtS exacts that the underlying signature satisfies the highest security notion, i.e. strong unforgeability under chosen message attacks. Such a need is justified by the possibility, in case the signature scheme does not satisfy this requirement, to create a new signcryption on any message given one signcryption on it (just generate a new digital signature on the encryption e), which entitles the indistinguishability adversary to retrieve the message in the game described in Definition 6.

The new StE paradigm, described in Subsection 3.2, does not suffer the recourse to stronger assumptions on the underlying signature. It further provides anonymity of the sender; the signcryption on a message m is a ciphertext, whereas in the EtS paradigm, everyone can check whether the sender was involved in a signcryption (e, s) by checking the validity of the digital signature (using the sender's public key) on the ciphertext e. However, verifiability turns out to be a hurdle; StE applies the signing algorithm (of the used signature scheme) to the message to be signcrypted concatenated with the used encapsulation. It further produces an encryption of the resulting signature concatenated with the message in question. As we are interested in proving the validity of the produced signcryption, we will need to exploit the homomorphic properties of the signature and of the encryption schemes in order to provide proofs of knowledge of the encrypted signature and message. As a consequence, the used encryption and signature schemes need to operate on elements from a set with a known algebraic structure rather than on bit-strings.

To sum-up, EtS provides efficient verifiability but at the expense of the sender's anonymity, and of the security requirements on the building blocks. StE achieves better privacy using cheap constituents but at the expense of verifiability. It would be nice to have a technique that combines the merits of both paradigms while avoiding their drawbacks. This is the main contribution in this section; the core of the idea consists in first encrypting the message to be signcrypted using a public key encryption scheme, then applying the StE paradigm to the produced encryption. The result of this operation in addition to the encrypted message form the new signcryption of the message in question. In other terms, this technique can be seen as a merge between the EtS and the StE paradigms; thus we can term it the "encrypt-then-sign-then-encrypt" paradigm (EtStE).

5.1 The construction

Setup. Consider a signature scheme Σ, an encryption scheme Γ, and another encryption scheme $(\mathcal{K}, \mathcal{D})$ derived from the KEM/DEM paradigm. Next, on input the security parameter $\kappa = (\kappa_1, \kappa_2, \kappa_3)$, generate the parameters param of these schemes. We assume that signatures issued with Σ can be written as (r, s), where r reveals no information about the signed message nor about the public signing key, and s represents the "significant" part of the signature.

Key generation. On input the security parameter κ and the public parameters param, invoke the key generation algorithms of the building blocks and set the sender's key pair to $(\Sigma.\mathsf{pk}, \Sigma.\mathsf{sk})$, and the receiver's key pair to $(\{\Gamma.\mathsf{pk}, \mathcal{K}.\mathsf{pk}\}, \{\Gamma.\mathsf{sk}, \mathcal{K}.\mathsf{sk}\})$.

Signcrypt. On a message m, produce an encryption $e = \Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(m)$ of m. Then fix a key k along with its encapsulation c using $\mathcal{K}.\mathsf{encrypt}_{\mathcal{K}.\mathsf{pk}}$, produce a signature (r, s) on $c \| e$, and finally encrypt s with k using $\mathcal{D}.\mathsf{encrypt}$. The signcryption of m is the tuple $(e, c, \mathcal{D}.\mathsf{encrypt}_k(s), r)$.

Prove Validity. Given a signcryption $\mu = (\mu_1, \mu_2, \mu_3, \mu_4)$ on a message m, the prover proves knowledge of the decryption of μ_1, and of the decryption of (μ_2, μ_3), which together with μ_4 forms a valid digital signature on $\mu_2 \| \mu_1$. The private input is either the randomness used to create μ or $\{\Gamma.\mathsf{sk}, \mathcal{K}.\mathsf{sk}\}$.

Unsigncrypt. On a signcryption a $(\mu_1, \mu_2, \mu_3, \mu_4)$, compute $m = \Gamma.\mathsf{decrypt}_{\Gamma.\mathsf{sk}}(\mu_1)$ and $k = \mathcal{K}._{\mathcal{K}.\mathsf{sk}}(\mu_2)$. Check whether $(\mathcal{D}.\mathsf{decrypt}_k(\mu_3), \mu_4)$ is valid signature on $\mu_2 \| \mu_1$; if yes then output m, otherwise output $\perp$.

Confirm/Deny. On input a putative signcryption $\mu = (\mu_1, \mu_2, \mu_3, \mu_4)$ on a message m, use the receiver's private key to prove that m is/isn't the decryption of μ_1, and prove knowledge of the decryption of (μ_2, μ_3), which together with μ_4 forms a valid/invalid digital signature on $\mu_2 \| \mu_1$.

Public Verify. On a valid signcryption $\mu = (\mu_1, \mu_2, \mu_3, \mu_4)$ on a message m, output a ZK non-interactive proof that μ_1 encrypts m, in addition to $(\mathcal{D}.\mathsf{decrypt}_{\mathcal{K}.\mathsf{decap}(\mu_2)}(\mu_3), \mu_4)$.

5.2 Analysis

Theorem 11 *Given $(t, q_s) \in \mathbb{N}^2$ and $\varepsilon \in [0, 1]$, 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 12 *Given $(t, q_s, q_v, q_u, q_{cd}, q_{pv}) \in \mathbb{N}^6$ and $(\varepsilon, \epsilon') \in [0,1]^2$, the construction proposed above is $(t, \epsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$-IND-CCA secure if it uses a (t, ϵ', q_s)-EUF-CMA secure digital signature, an IND-CPA secure encryption, an IND-OT secure DEM, and a $(t + q_s(q_u + q_{cd} + q_{pv}), \frac{\epsilon(1-\epsilon')^{q_{cd}+q_u+q_{pv}}}{2})$-IND-CPA secure KEM.*

We provide both proofs in Appendix E.1 and Appendix E.2 respectively.

Our aim in the rest of this paragraph consists in identifying suitable classes of encryption/signature schemes that renders the $\mathsf{proveValidity}$ and $\{\mathsf{confirm}, \mathsf{deny}\}$ efficient. These protocols comprise the following sub-protocols:

1. Proving knowledge of the decryption of a ciphertext produced using the encryption scheme Γ.
2. Proving that a message is/isn't the decryption of a certain ciphertext produced using Γ.
3. Proving knowledge of the decryption of a ciphertext produced using $(\mathcal{K}, \mathcal{D})$, and that this decryption forms a valid/invalid digital signature, issued using Σ, on some known string.

It is natural to instantiate the encryption scheme Γ from the class $\mathbb{E}$ defined in Definition 3. The first two sub-protocols can be efficiently carried out using the proofs depicted in Figure 1 and Figure 2. For the last sub-protocol, one can consider encryption schemes from the class $\mathbb{E}$ that are derived from the KEM/DEM paradigm, in addition to signature schemes that accept efficient proofs of knowledge. We provide in Appendix E.3 the class of used signatures as well as the proof underlying the third sub-protocol.

5.3 Extension to Multi-user security

The above EtStE technique can be extended to achieve security in the multi-user setting, as defined in Subsection B, by applying the standard techniques [1, 39]. More specifically, one considers a tag-based encryption scheme Γ, a tag-based KEM $\mathcal{K}$, a DEM $\mathcal{D}$, an a signature scheme. The sender's key pair is the signature scheme key pair, whereas the receiver's key pair comprise both key pairs of Γ and $\mathcal{K}$. Signcryption on a message m w.r.t. a sender's public key $\Sigma.\mathsf{pk}$ and a receiver's public key $(\Gamma.\mathsf{pk}, \mathcal{K}.\mathsf{pk})$ is generated as follows. First compute an encryption e on m (with Γ) w.r.t. the tag $\Sigma.\mathsf{pk}$, then generate a key k and its encapsulation c w.r.t. the same tag (with $\mathcal{K}$), then compute a digital signature on $c\|e\|\{\Gamma.\mathsf{pk}, \mathcal{K}.\mathsf{pk}\}$, and finally sign the "significant" part of this signature using k. The signcryption consists of the result of this encryption, the remaining part of the signature, and (e, c). The rest is similar to the paradigm in the two-user setting.

Theorem 13 *Given $(t, q_s) \in \mathbb{N}^2$ and $\varepsilon \in [0,1]$, the above construction is (t, ϵ, q_s)-dM-EUF-CMA secure if the underlying digital signature scheme is (t, ϵ, q_s)-EUF-CMA secure.*

The proof is similar to that of Theorem 9 and of Theorem 11. □

Theorem 14 *Given $(t, q_s, q_v, q_u, q_{cd}, q_{pv}) \in \mathbb{N}^6$ and $(\varepsilon, \epsilon') \in [0,1]^2$, the above construction is $(t, \epsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$-fM-IND-CCA secure if it uses a (t, ϵ', q_s)-EUF-CMA secure digital signature, an IND-sTag-CCA secure encryption, an IND-OT secure DEM, and a $(t + q_s(q_u + q_{cd} + q_{pv}), \frac{\epsilon(1-\epsilon')^{q_{cd}+q_u+q_{pv}}}{2}, q_{cd} + q_u + q_{pv})$-IND-sTag-CCA secure KEM.*

The proof is similar to that of Theorem 9 and of Theorem 12.

□

6 Conclusion

We provided a model for verifiable signcryption which captures many real-life applications of this primitive. We further studied the classical generic constructions of signcryption and provided optimzations of these when they fail to provide secure and efficient instantiations. The resulting schemes have insider unforgeability and outsider indistinguishability. Achieving insider indistinguishability is a natural aspiration. Unfortunately, it does not seem plausible without resting on IND-CCA secure encryption, which impacts negatively verifiability as we can no longer use homomorphic encryption. Another enhancement of our schemes would be to get rid of interaction in the used protocols and use instead non-interactive (designated verifier) proofs. This step does not seem so immediate as it is known that non-interactive proofs are in general difficult to obtain in the standard model.

References

1. J. H. An, Y. Dodis, and T. Rabin, *On the Security of Joint Signature and Encryption.*, Advances in Cryptology - EUROCRYPT 2002 (L. R. Knudsen, ed.), LNCS, vol. 2332, Springer, 2002, pp. 83–107.
2. J. Baek, R. Steinfeld, and Y. Zheng, *Formal Proofs for the Security of Signcryption*, J. Cryptology **20** (2007), no. 2, 203–235.
3. F. Bao and R. H. Deng, *A Signcryption Scheme with Signature Directly Verifiable by Public Key*, Public Key Cryptography (H. Imai and Y. Zheng, eds.), LNCS, vol. 1431, Springer, 1998, pp. 55–59.
4. 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.
5. M. Bellare and P. Rogaway, *The Exact Security of Digital Signatures: How to Sign with RSA and Rabin.*, Advances in Cryptology - EUROCRYPT'96 (U. M. Maurer, ed.), LNCS, vol. 1070, Springer, 1996, pp. 399–416.
6. M. Blum, P. Feldman, and S. Micali, *Non-Interactive Zero-Knowledge and Its Applications (Extended Abstract)*, STOC (J. Simon, ed.), ACM Press, 1988, pp. 103–112.
7. 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.
8. D. Boneh, X. Boyen, and H. Shacham, *Short Group Signatures.*, in Franklin [25], pp. 41–55.
9. D. Boneh, B. Lynn, and H. Shacham, *Short Signatures from the Weil Pairing.*, J. Cryptology **17** (2004), no. 4, 297–319.
10. G. Brassard, D. Chaum, and C. Crépeau, *Minimum disclosure proofs of knowledge.*, J. Comput. Syst. Sci. **37** (1988), no. 2, 156–189.
11. E. Bresson and J. Stern, *Proofs of Knowledge for Non-Monotone Discrete-Log Formulae and Applications.*, Information Security, ISC 2002 (A. H. Chan and V. D. Gligor, eds.), LNCS, vol. 2433, Springer, 2002, pp. 272–288.
12. 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.
13. ______, *Signature Schemes and Anonymous Credentials from Bilinear Maps*, in Franklin [25], pp. 56–72.
14. 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.
15. 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.
16. D. Chiba, T. Matsuda, J. C. N. Schuldt, and K. Matsuura, *Efficient Generic Constructions of Signcryption with Insider Security in the Multi-user Setting*, ACNS (J. Lopez and G. Tsudik, eds.), LNCS, vol. 6715, 2011, pp. 220–237.
17. S. M. Chow, S-M. Yiu, L. Hui, and K. P. Chow, *Efficient Forward and Provably Secure ID-Based Signcryption Scheme with Public Verifiability and Public Ciphertext Authenticity*, ICISC (J. I. Lim and D. H. Lee, eds.), LNCS, vol. 2971, Springer, 2003, pp. 352–369.
18. S. S. M. Chow and K. Haralambiev, *Non-interactive Confirmer Signatures*, CT-RSA (A. Kiayias, ed.), LNCS, vol. 6558, Springer, 2011, pp. 49–64.
19. R. Cramer and V. Shoup, *Signature schemes based on the strong RSA assumption.*, ACM Trans. Inf. Syst. Secur. **3** (2000), no. 3, 161–185.
20. ______, *Design and Analysis of Practical Public-Key Encryption Schemes Secure Against Adaptive Chosen Ciphertext Attack.*, SIAM J. Comput. **33** (2003), no. 1, 167–226.
21. I. Damgård, N. Fazio, and A. Nicolosi, *Non-interactive zero-knowledge from homomorphic encryption*, in Halevi and Rabin [31], pp. 41–59.
22. A. W. Dent, *Hybrid Signcryption Schemes with Outsider Security*, ISC (J. Zhou, J. Lopez, R. H. Deng, and F. Bao, eds.), LNCS, vol. 3650, Springer, 2005, pp. 203–217.
23. L. El Aimani, *On Generic Constructions of Designated Confirmer Signatures*, in Roy and Sendrier [43], Full version available at the Cryptology ePrint Archive, Report 2009/403, pp. 343–362.
24. T. El Gamal, *A Public Key Cryptosystem and a Signature Scheme based on Discrete Logarithms.*, IEEE Trans. Inf. Theory **31** (1985), 469–472.
25. 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.
26. R. Gennaro, S. Halevi, and T. Rabin, *Secure Hash-and-Sign Signatures Without the Random Oracle.*, in Stern [48], pp. 397–416.
27. O. Goldreich, *Foundations of cryptography. Basic Tools.*, Cambridge University Press., 2001.
28. 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.
29. 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.
30. J. Groth and A. Sahai, *Efficient Non-interactive Proof Systems for Bilinear Groups*, EUROCRYPT 2008 (N. P. Smart, ed.), LNCS, vol. 4965, Springer, 2008, pp. 415–432.

31. Shai Halevi and Tal Rabin (eds.), *Theory of cryptography, third theory of cryptography conference, tcc 2006, new york, ny, usa, march 4-7, 2006, proceedings*, LNCS, vol. 3876, Springer, 2006.
32. S-H. Heng and K. Kurosawa (eds.), *The fourth international conference on provable security*, LNCS, vol. 6402, Springer, 2010.
33. 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.
34. I. Jeong, H. Jeong, H. Rhee, D. Lee, and J. Lim, *Provably Secure Encrypt-then-Sign Composition in Hybrid Signcryption*, in Lee and Lim [36], pp. 16–34.
35. E. Kiltz, *Chosen-Ciphertext Security from Tag-Based Encryption*, in Halevi and Rabin [31], pp. 581–600.
36. P. J. Lee and C. H. Lim (eds.), *Information security and cryptology* - ICISC *2002, 5th international conference seoul, korea, november 28-29, 2002, revised papers*, LNCS, vol. 2587, Springer, 2003.
37. M. Liskov and S. Micali, *Online-Untransferable Signatures*, Public Key Cryptography (R. Cramer, ed.), LNCS, vol. 4939, Springer, 2008, pp. 248–267.
38. C. Ma, *Efficient Short Signcryption Scheme with Public Verifiability*, Inscrypt (H. Lipmaa, M. Yung, and D. Lin, eds.), LNCS, vol. 4318, Springer, 2006, pp. 118–129.
39. T. Matsuda, K. Matsuura, and J. Schuldt, *Efficient Constructions of Signcryption Schemes and Signcryption Composability*, in Roy and Sendrier [43], pp. 321–342.
40. P. Paillier, *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes*, in Stern [48], pp. 223–238.
41. 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.
42. D. Pointcheval and J. Stern, *Security Arguments for Digital Signatures and Blind Signatures.*, J. Cryptology **13** (2000), no. 3, 361–396.
43. B. Roy and N. Sendrier (eds.), *Progress in cryptology* - INDOCRYPT *2009*, vol. 5922, Berlin, Heidelberg, 2009.
44. C. P. Schnorr, *Efficient signature generation by smart cards.*, J. Cryptology **4** (1991), no. 3, 161–174.
45. S. Selvi, S. Vivek, and P. Pandu Rangan, *Identity Based Public Verifiable Signcryption Scheme*, in Heng and Kurosawa [32], pp. 244–260.
46. J-B Shin, K. Lee, and K. Shim, *New DSA-Verifiable Signcryption Schemes*, in Lee and Lim [36], pp. 35–47.
47. V. Shoup and R. Gennaro, *Securing Threshold Cryptosystems against Chosen Ciphertext Attack*, J. Cryptology **15** (2002), no. 2, 75–96.
48. 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.
49. 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.
50. 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.
51. Y. Zheng, *Digital Signcryption or How to Achieve Cost(Signature & Encryption)* $\ll$ *Cost(Signature) + Cost(Encryption).*, Advances in Cryptology - CRYPTO'97 (B. S. Kaliski Jr., ed.), LNCS, vol. 1294, Springer, 1997, pp. 165–179.

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 proveValidity. The standard security notion for a signature scheme is existential unforgeability under chosen message attacks (EUF-CMA), which was introduced in [29]. 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 to 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 $\in$ {OW, IND, NM} and ATK $\in$ {CPA, PCA, CCA}. We refer to [4] for the formal definitions of these notions as well as for the relations they satisfy.

A.3 Tag-based encryption

Tag-based encryption, also referred to as encryption with labels, was first introduced in [47]. In these schemes, the encryption algorithm takes as input, in addition to the public key pk and the message m intended to be encrypted, a tag t which specifies information related to the message m and its encryption context. Similarly, the decryption algorithm takes additionally to the ciphertext and the private key the tag under which the ciphertext was created. Security notions are then defined as usual except that the adversary specifies to his challenger the tag to be used in the challenge ciphertext, and in case he (the adversary) is allowed to query oracles, then he cannot query them on the pair formed by the challenge ciphertext and the tag used to form it. There are also weakened security models for this type of encryption where the adversary specifies the challenge tag before getting the parameters of the scheme, and during the game, he (the adversary) is not allowed to query the allowed oracles w.r.t. the challenge tag; we talk in this case about selective tag security. We specify in the following the formal definition of IND-sTag-CCA security for tag-based encryption:

Definition 4 (IND-sTag-CCA indistinguishability). *Let Γ be a tag-based encryption scheme scheme. Let further $\mathcal{A}$ denote a PPTM. We consider the following random experiment:*

$$\boxed{\textit{Experiment } \mathbf{Exp}^{\textsf{\textit{ind-sTag-CCA}}-b}_{\Gamma,\mathcal{A}}(1^\kappa)}$$

$$\begin{array}{l}
(t^\star, \mathcal{I}_0) \leftarrow \mathcal{A}(1^\kappa, \mathsf{init}) \\
(\mathsf{pk}, \mathsf{sk}) \leftarrow \Gamma.\mathsf{keygen}(1^\kappa), \\
(m_0^\star, m_1^\star, \mathcal{I}) \leftarrow \mathcal{A}^{\mathfrak{O}}(\textsf{\textit{find}}, \mathsf{pk}, \mathcal{I}_0) \\
\qquad\qquad\qquad\qquad \big| \, \mathfrak{O} : (c, t)[t \neq t^\star] \longmapsto \Gamma.\mathsf{decrypt}_{\mathsf{sk}}(c, t) \\
c^\star \leftarrow \Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m_b^\star, t^\star) \\
d \leftarrow \mathcal{A}^{\mathfrak{O}}(\textsf{\textit{guess}}, \mathcal{I}, c^\star, t^\star) \\
\\
\textit{Return } d
\end{array}$$

We define the advantage *of $\mathcal{A}$* via*:*

$$\mathbf{Adv}^{\textsf{\textit{ind-sTag-CCA}}}_{\Gamma,\mathcal{A}}(1^\kappa) = \left| \Pr\left[\mathbf{Exp}^{\textsf{\textit{ind-sTag-CCA}}-b}_{\Gamma,\mathcal{A}}(1^\kappa) = b \right] - \frac{1}{2} \right|.$$

Given $(t, q_d) \in \mathbb{N}^2$ and $\varepsilon \in [0, 1]$, $\mathcal{A}$ is called a (t, ε, q_d)-ind-sTag-CCA adversary against Γ if, running in time t and issuing q_d decryption queries, $\mathcal{A}$ has $\mathbf{Adv}^{\mathsf{ind-sTag-CCA}}_{\Gamma,\mathcal{A}}(1^\kappa) \geq \varepsilon$. The scheme Γ is said to be (t, ε, q_d)-ind-sTag-CCA secure if no (t, ε, q_d)-ind-sTag-CCA adversary against it exists.

A.4 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 [33] 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 an *indistinguishable under a one time attack (IND-OT)* DEM.

A.5 Commitment schemes

A commitment scheme [10] consists of the following algorithms:

- setup: the setup algorithm that generates the public parameters of the system.
- keygen: generates probabilistically a public commitment key pk.
- commit: a probabilistic algorithm that, on input a public key pk and a message m, produces a pair (c, r): c serves as the commitment value (locked box), and r as the opening value.
- open: this is a deterministic algorithm that given a commitment (c, r), w.r.t. a public key pk, on a alleged message m, checks whether $c \stackrel{?}{=} \mathsf{commit}_{\mathsf{pk}}(m, r)$.

The algorithm open must succeed if the commitment was correctly formed (correctness). Moreover, we require the following security properties:

1. **Hiding.** It is hard for an adversary A to generate two messages m_0, m_1 such that he can distinguish between their corresponding locked boxes c_o, c_1. That is, c reveals no information about m.
   A commitment scheme is (t, ϵ)-hiding if no adversary A, operating in time t, can succeed in the above game with probability greater than ϵ, where the probability is taken over the random coins of both A and his challenger.
2. **Binding.** It is hard for an adversary A to come up with a *collision* (c, d, d') such that (c, d) and (c, d') are valid commitments for m and m' resp and $m \neq m'$.

B Multi-User Security

Definition 5 (Multi-user Unforgeability). *We consider a signcryption scheme* sc *given by the algorithms/protocols defined earlier in this document. Let $\mathcal{A}$ be a PPTM. We consider the following random experiment:*

$$\boxed{\textit{Experiment } \mathbf{Exp}^{\textit{dm-euf-cma}}_{\mathsf{sc},\mathcal{A}}(1^\kappa)}$$

$\mathsf{param} \leftarrow \mathsf{sc.setup}(1^\kappa)$
$(\mathsf{sk}^\star_S, \mathsf{pk}^\star_S) \leftarrow \mathsf{sc.keygen}_S(1^\kappa, \mathsf{param})$
$(\mathsf{sk}^\star_R, \mathsf{pk}^\star_R, \mu^\star) \leftarrow \mathcal{A}^{\mathfrak{S}}(\mathsf{pk}^\star_S)$
$\qquad\qquad \mathfrak{S} : (m, \mathsf{sk}_R, \mathsf{pk}_R) \longmapsto \mathsf{sc.signcrypt}\{\mathsf{sk}^\star_S, \mathsf{pk}_S, \mathsf{pk}_R\}(m)$
return 1 if and only if the following properties are satisfied:
- $\mathsf{sc.unsigncrypt}_{\{\mathsf{sk}^\star_R, \mathsf{pk}^\star_R, \mathsf{pk}^\star_S\}}[\mu^\star] = m^\star$
- $m^\star$ *was not queried to* $\mathfrak{S}$ *w.r.t.* $\mathsf{pk}^\star_R$.

We define the success *of* $\mathcal{A}$ via*:*

$$\mathbf{Succ}^{\textit{dm-euf-cma}}_{\mathsf{sc},\mathcal{A}}(1^\kappa) = \Pr\left[\mathbf{Exp}^{\textit{dm-euf-cma}}_{\mathsf{sc},\mathcal{A}}(1^\kappa) = 1\right].$$

Given $(t, q_s) \in \mathbb{N}^2$ *and* $\varepsilon \in [0, 1]$, $\mathcal{A}$ *is called a* (t, ε, q_s)-*dM-EUF-CMA (dynamic Multi-user Existentially Unforgeable against Chosen Message Attacks) adversary against* sc *if, running in time* t *and issuing* q_s *queries to the* sc.signcrypt *oracle,* $\mathcal{A}$ *has* $\mathbf{Succ}^{\textit{dm-euf-cma}}_{\mathsf{sc},\mathcal{A}}(1^\kappa) \geq \varepsilon$. *The scheme* sc *is said to be* (t, ε, q_s)-*dM-EUF-CMA secure if no* (t, ε, q_s)-*dM-EUF-CMA adversary against it exists.*

Remark 5. A tangible difference between unforgeability in the multi-user setting and unforgeability in the two-user setting lies in the possibility of the adversary in the former to return a forgery on a message $m^\star$ that may have been queried before but w.r.t. a receiver's key different from the returned receiver's key $\mathsf{pk}_R^\star$.

Definition 6 (Multi-user Indistinguishability). *Let* sc *be a signcryption scheme, and let $\mathcal{A}$ be a PPTM. We consider the following random experiment for $b \xleftarrow{R} \{0,1\}$:*

$$\boxed{\textit{Experiment } \mathbf{Exp}_{\mathsf{sc},\mathcal{A}}^{\textit{fM-ind-cca-b}}(1^\kappa)}$$

$$
\begin{array}{l}
\mathsf{param} \leftarrow \mathsf{sc.setup}(1^\kappa) \\
(\mathsf{sk}_S^\star, \mathsf{pk}_S^\star) \leftarrow \mathsf{sc.keygen}_S(1^\kappa, \mathsf{param}) \\
(\mathsf{sk}_R^\star, \mathsf{pk}_R^\star) \leftarrow \mathsf{sc.keygen}(1^\kappa, \mathsf{param}) \\
(m_0^\star, m_1^\star, \mathcal{I}) \leftarrow \mathcal{A}^{\mathfrak{S},\mathfrak{V},\mathfrak{U},\mathfrak{C}}(\textit{find}, \mathsf{pk}_S^\star, \mathsf{pk}_R^\star) \\
\qquad \left| \begin{array}{l}
\mathfrak{S} : m \longmapsto \mathsf{sc.signcrypt}_{\{\mathsf{sk}_S^\star, \mathsf{pk}_S^\star, \mathsf{pk}_R^\star\}}(m) \\
\mathfrak{V} : \mu \longmapsto \mathsf{sc.proveValidity}(\mu, \mathsf{pk}_S^\star, \mathsf{pk}_R^\star) \\
\mathfrak{U} : \mu \longmapsto \mathsf{sc.unsigncrypt}_{\mathsf{sk}_R^\star, \mathsf{pk}_R^\star, \mathsf{pk}_S^\star}(\mu) \\
\mathfrak{C} : (\mu, m) \longmapsto \mathsf{sc.}\{\mathsf{confirm}, \mathsf{deny}\}(\mu, m, \mathsf{pk}_R^\star, \mathsf{pk}_S^\star) \\
\mathfrak{P} : (\mu, m) \longmapsto \mathsf{sc.publicVerify}(\mu, m, \mathsf{pk}_R^\star, \mathsf{pk}_S^\star)
\end{array} \right. \\
\mu^\star \leftarrow \mathsf{sc.signcrypt}_{\{\mathsf{sk}_S^\star, \mathsf{pk}_S^\star, \mathsf{pk}_R^\star\}}(m_b^\star) \\
d \leftarrow \mathcal{A}^{\mathfrak{S},\mathfrak{V},\mathfrak{U},\mathfrak{C}}(\textit{guess}, \mathcal{I}, \mu^\star, \mathsf{pk}_S^\star, \mathsf{pk}_R^\star) \\
\quad \left| \begin{array}{l}
\mathfrak{S} : m \longmapsto \mathsf{sc.signcrypt}_{\{\mathsf{sk}_S^\star, \mathsf{pk}_S^\star, \mathsf{pk}_R^\star\}}(m) \\
\mathfrak{V} : \mu \longmapsto \mathsf{sc.proveValidity}(\mu, \mathsf{pk}_S^\star, \mathsf{pk}_R^\star) \\
\mathfrak{U} : \mu (\neq \mu^\star) \longmapsto \mathsf{sc.unsigncrypt}_{\mathsf{sk}_R^\star, \mathsf{pk}_R^\star, \mathsf{pk}_S^\star}(\mu) \\
\mathfrak{V} : (\mu, m)(\neq (\mu^\star, m_i^\star), i = 0, 1) \longmapsto \mathsf{sc.}\{\mathsf{confirm}, \mathsf{deny}\}(\mu, m, \mathsf{pk}_R^\star, \mathsf{pk}_S^\star) \\
\mathfrak{P} : (\mu, m)(\neq (\mu^\star, m_i^\star), i = 0, 1) \longmapsto \mathsf{sc.publicVerify}(\mu, m, \mathsf{pk}_R^\star, \mathsf{pk}_S^\star)
\end{array} \right.
\end{array}
$$

Return d

We define the advantage *of $\mathcal{A}$* via*:*

$$\mathbf{Adv}_{\mathsf{sc},\mathcal{A}}^{\textit{fm-ind-cca}}(1^\kappa) = \left| \Pr\left[\mathbf{Exp}_{\mathsf{sc},\mathcal{A}}^{\textit{fm-ind-cca-b}}(1^\kappa) = b \right] - \frac{1}{2} \right|.$$

Given $(t, q_s, q_v, q_u, q_{cd}, q_{pv}) \in \mathbb{N}^6$ and $\varepsilon \in [0,1]$, $\mathcal{A}$ is called a $(t, \varepsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$-fM-IND-CCA (fixed challenge Multi-user Indistinguishable under Chosen Ciphertext Attacks) adversary against sc *if, running in time t and issuing q_s queries to the* sc.signcrypt *oracle, q_v queries to the* sc.proveValidity *oracle, q_u queries to the* sc.unsigncrypt *oracle, q_{cd} queries to the* sc.{confirm, deny} *oracle , and q_{pv} queries to the* sc. *oracle, $\mathcal{A}$ has $\mathbf{Adv}_{\mathsf{sc},\mathcal{A}}^{\mathsf{ind-cca}}(1^\kappa) \geq \varepsilon$. The scheme* sc *is said to be $(t, \varepsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$-fM-IND-CCA secure if no $(t, \varepsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$-fM-IND-CCA adversary against it exists.*

Remark 6. Note that in the Multi-user setting, the indistinguishability adversary is allowed to ask the unsigncryption of the challenge w.r.t. any receiver's key except that of the target receiver.

C Analysis of the StE and the CtEaS Paradigms

C.1 Proof of Lemma 2

Proof. Let $\mathcal{R}$ be the key-preserving reduction that reduces NM-CPA breaking the encryption scheme underlying the construction to IND-CCA breaking the construction (from the StE or EtS) itself. We will construct an algorithm $\mathcal{M}$ that uses $\mathcal{R}$ to NM-CPA break the same encryption scheme by simulating an execution of the IND-CCA adversary $\mathcal{A}$ against the construction.

Let Γ be the encryption scheme $\mathcal{M}$ is trying to attack.

– *StE paradigm.* $\mathcal{M}$ (behaving as $\mathcal{A}$) will query $\mathcal{R}$ on two messages m_0, m_1 ($m_0 \neq m_1$) for signcryption. Let μ_0, μ_1 be the corresponding signcryptions respectively. $\mathcal{M}$ will query again (μ_i, m_i), $i \in \{0,1\}$, for public verification. Let σ_0, σ_1 be the corresponding answers respectively. At that point, $\mathcal{M}$ outputs $D = \{m_0 \| \sigma_0, m_1 \| \sigma_1\}$, to his NM-CPA challenger, as a distribution probability from which the messages will be drawn. He gets in response a challenge encryption $\mu^\star$, of either $m_0 \| \sigma_0$ or $m_1 \| \sigma_1$ 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 solve his task, $\mathcal{M}$ queries $\mu^\star$ for unsigncryption. Let m_b be the result of such a query with $b \in \{0,1\}$. $\mathcal{M}$ will output $\Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(\overline{m_b \| \sigma_b})$ ($\overline{m}$ refers to the bit-complement of the element m) and the relation R: $R(m, m') = (m' = \overline{m})$. Finally $\mathcal{M}$ aborts the game (stops simulating an IND-CCA attacker against the generic construction).
– *CtEaS paradigm.* Similarly, $\mathcal{M}$ queries $\mathcal{R}$ on m_0, m_1 ($m_0 \neq m_1$) for signcryption. Let $\mu_0 = (e_0, c_0, \sigma_0)$ and $\mu_1 = (e_1, c_1, \sigma_1)$ be the corresponding signcryptions. $\mathcal{M}$ will then query μ_0, μ_1, along with the corresponding messages, for public verification. Let $m_0 \| r_0$ and $m_1 \| r_1$ be the corresponding answers. $\mathcal{M}$ will output $D = \{m_0 \| r_0, m_1 \| r_1\}$, to his NM-CPA challenger, as a distribution probability from which the messages will be drawn. He will receive a challenge encryption $e^\star$, of either $m_0 \| r_0$ or $m_1 \| r_1$. $\mathcal{M}$ will then query the unsigncryption of $(e^\star, c_b, \sigma_b)$ for some $b \xleftarrow{R} \{0,1\}$. If the answer is different from $\perp$, then $e^\star$ is indeed an encryption of $m_b \| r_b$, and thus $\mathcal{M}$ will output $\Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(\overline{m_b \| r_b})$ and the relation R (defined above). Otherwise, $\mathcal{M}$ will output $\Gamma.\mathsf{encrypt}_{\Gamma.\mathsf{pk}}(\overline{m_{1-b} \| r_{1-b}})$ and the same relation R. Finally $\mathcal{M}$ aborts the game (stops simulating an IND-CCA attacker against the generic construction).

□

C.2 Generalization to Arbitrary Reductions

Non malleable key generators We define the *non malleability of a cryptosystem key generator* through the following two games:

1. 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 (w.r.t. $\Gamma.\mathsf{pk}$). 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 $\mathcal{A}$. Note that the goal of **Game 0** is to include all key-preserving reductions $\mathcal{R}$ from NM-CPA (or OW-CCA) breaking the cryptosystem in question to solving a problem A, which is reducible to OW-CPA breaking the same cryptosystem.
2. 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 of the OW-CPA oracle.

Definition 7. *A cryptosystem Γ is said to have a non malleable key generator if*
$\Delta = max_{\mathcal{A}} |\mathsf{succ}_{\Gamma}^{\mathsf{Game1}}(\mathcal{A})\text{-}\mathsf{succ}_{\Gamma}^{\mathsf{Game0}}(\mathcal{A})|$ *is negligible 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 $\mathsf{pk}' \neq \mathsf{pk}$.

To generalize the impossibility results in Subsection 3.1, we first need the following Lemma (similar to Lemma 6 of [41])

Lemma 3. *Let $\mathcal{A}$ be an adversary solving a problem A, reducible to OW-CPA breaking a cryptosystem Γ, and let $\mathcal{R}$ be an arbitrary reduction $\mathcal{R}$ that NM-CPA (OW-CCA) breaks a cryptosystem Γ, given access to $\mathcal{A}$. We have* $\mathsf{adv}(\mathcal{R}) \leq \mathsf{succ}_{\Gamma}^{\mathsf{Game1}}(\mathcal{A})$

Proof. We will construct an algorithm $\mathcal{M}$ that plays **Game 1** with respect to a perfect oracle for $\mathcal{A}$ and succeeds in breaking the NM-CPA (OW-CCA) security of Γ with the same success probability of $\mathcal{R}$. 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 answer the queries. In fact, by assumption, the problem A is reducible to OW-CPA solving Γ. Finally, when $\mathcal{R}$ outputs the result to $\mathcal{M}$, the latter will output the same result to his own challenger. □

Theorem 1. *If the cryptosystem underlying the StE or the CtEaS constructions has a non malleable key generator and is OW-CCA (NM-CPA) secure, then, there is no efficient reduction which reduces OW-CCA (NM-CPA) breaking the cryptosystem to IND-CCA breaking the construction.*

Proof. We first remark that the indistinguishability of constructions from the plain StE and CtEaS paradigms is perfectly reducible to OW-CPA breaking the cryptosystem underlying the construction.
Next, we note that the advantage of the meta-reduction $\mathcal{M}$ in the proof of Lemma 1 (Lemma 2) is the same as the advantage of any key-preserving reduction $\mathcal{R}$ reducing the indistinguishability of StE and CtEaS constructions to the NM-CPA (OW-CCA) security of its underlying cryptosystem Γ. For instance, this applies to the reduction making the best use of an indistinguishability $\mathcal{A}$ against the constructions. Therefore we have:

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

where $\mathsf{succ}(NM\text{-}CPA[\Gamma])$ ($\mathsf{succ}(OW\text{-}CCA[\Gamma])$) is the success of breaking Γ in the NM-CPA (OW-CCA) sense.
Now, Let $\mathcal{R}$ be an arbitrary reduction from NM-CPA (OW-CCA) breaking a cryptosystem Γ, with a non malleable key generator, to IND-CCA breaking the (StE and CtEaS) constructions (using the same cryptosystem Γ):

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

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

C.3 A breach in indistinguishability with homomorphic encryption

Definition 8 (Homomorphic encryption). *A homomorphic public encryption scheme* $\Gamma = (\Gamma.\mathsf{keygen}, \Gamma.\mathsf{encrypt}, \Gamma.\mathsf{decrypt})$ *has the following properties:*

1. *The message space $\mathcal{M}$ and the ciphertext space $\mathcal{C}$ are groups w.r.t. some binary operations $*$ and $\circ_e$ respectively.*
2. $\forall m, m' \in \mathcal{M}$, $\forall(\mathsf{sk}, \mathsf{pk}) \leftarrow \Gamma.\mathsf{keygen}(1^\kappa)$ *for some security parameter κ:*

$$\Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m * m') = \Gamma.\mathsf{encrypt}_{\mathsf{pk}}(m) \circ_e \Gamma.\mathsf{encrypt}_{pk}(m').$$

Examples of homomorphic encryptions in the literature include EL GAMAL [24], BONEH-BOYEN-SHACHAM [8], and PAILLIER [40]. All those schemes are IND-CPA secure.

Fact 1 *The StE and the CtEaS paradigms cannot lead to IND-CCA secure constructions when used with homomorphic encryption.*

Proof. Let m_0, m_1 be the challenge messages the indistinguishability adversary $\mathcal{A}$ outputs to his challenger. We show that the mentioned paradigms are prone to these simple attacks if the underlying encryption scheme is homomorphic:

– *StE paradigm.* Let Γ and Σ denote respectively the encryption and signature schemes used as constituents. $\mathcal{A}$ will receive as a challenge signcryption a certain $\mu_b = \Gamma.\mathsf{encrypt}(\Sigma.\mathsf{sign}(m_b))$, where $b \in \{0,1\}$ and his task is to guess correctly the used b. To solve his challenge, $\mathcal{A}$ will obtain another encryption, say $\tilde{\mu_b}$, of $\mathsf{sign}(m_b)$ by multiplying μ_b with an encryption of the identity element (of the message space of Γ). According to Definition 6, $\mathcal{A}$ can query $\tilde{\mu_b}$ for unsigncryption and the answer to such a query is sufficient for $\mathcal{A}$ to conclude.

– *CtEaS paradigm.* Let Γ, Σ, and Ω denote respectively the encryption, signature, and commitment schemes used as building blocks. $\mathcal{A}$ will get as a challenge signcryption a certain $\mu_b = [\Gamma.\mathsf{encrypt}(m_b\|r), c = \Omega.\mathsf{commit}(m_b\|r), (c)]$ with $b \in \{0,1\}$. Similarly, $\mathcal{A}$ will compute a new signcyption of m_b by multiplying the first field of μ_b by an encryption of the identity element (of the message space of Γ). $\mathcal{A}$ will then query this new signcryption to the unsigncryption oracle, and the answer of the former is sufficient for $\mathcal{A}$ to solve his challenge.

□

Remark 7. Note that the EtS paradigm is resilient to the previous attack since the adversary would need to compute a valid digital signature on the newly computed encryption. This is not plausible in the game described in Definition 6.

C.4 Proof of Theorem 1

Proof. Let $\mathcal{A}$ be a (t, ϵ, q_s)-EUF-CMA adversary against the above signcryption scheme sc. We build a (t, ϵ, q_s)-EUF-CMA adversary, say $\mathcal{R}$, against the underlying signature scheme as follows.

setup **and** keygen**.** $\mathcal{R}$ gets the public key pk_S of the signature scheme from his challenger. He generates further the parameters of the KEM/DEM based encryption, namely $(\mathcal{K}.\mathsf{sk}, \mathcal{K}.\mathsf{pk})$, and hands then along with pk_S to $\mathcal{A}$ as settings for the target signcryption scheme sc.

$\mathsf{signcrypt}$. On a message m, $\mathcal{A}$ will generate a key k together with its encapsulation c using $\mathcal{K}.\mathsf{encap}$, then he will query his own challenger for a digital signature on $c\|m$. Upon receipt of the result of such a query, say σ, $\mathcal{R}$ will encrypt it in e using k, and will hand (c, e) as a signcryption of m. It is easy to see that such a simulation is indistinguishable from that of the standard algorithm described in Definition 1.

Final output. When $\mathcal{A}$ outputs his forgery $(c^\star, e^\star)$ on some message $m^\star$, $\mathcal{R}$ will output $\mathcal{D}.\mathsf{decrypt}_{k^\star}(e^\star)$, where $k^\star = \mathcal{K}.\mathsf{decap}_{\mathcal{K}.\mathsf{sk}}(c^\star)$, to his own challenger. In fact, $e^\star$ is an encryption of a valid digital signature, on $c^\star\|m^\star$, using the decapsulation of $c^\star$. It remains to show that $\mathcal{R}$ never requested his challenger for a digital signature on $c^\star\|m^\star$. Suppose that this the case, i.e. there exists an $1 \le i \le q_s$ such that $c^\star\|m^\star = c_i\|m_i$. this means that $m^\star = m_i$, since the encapsulations have the same bit-size, which contradicts the fact that $(c^\star, e^\star)$ is valid existential forgery on $m^\star$.

□

Remark 8. The previous theorem shows that existential unforgeability of the underlying digital signature scheme suffices to ensure existential unforgeability of the resulting construction. Actually, one can also show that this requirement on the digital signature guarantees also that no adversary, against the construction, can come up with a valid signcryption (c, e) (c is the encapsulation used to generate the signcryption μ) on a message m that has been queried before to the signing oracle but where c was never used to generate answers (signcryptions) to the signcryption queries.

To prove this claim, we construct from such an adversary, say $\mathcal{A}$, an EUF-CMA adversary $\mathcal{R}$ against the underlying digital signature scheme, which runs in the same time and has the same advantage as $\mathcal{A}$. In fact, $\mathcal{R}$ will simulate $\mathcal{A}$'s environment in the same way described in the proof of Theorem 1. When $\mathcal{A}$ outputs his forgery $\mu^\star = (c^\star, e^\star)$ on a message m_i that has been previously queried to the signing oracle, $\mathcal{R}$ decrypts $\mu^\star$ in $\sigma^\star$, which by definition forms a valid digital signature on $c^\star\|m_i$. Since by assumption $c^\star$ was never used to generate signcryptions on the queried messages, $\mathcal{R}$ never invoked his own challenger for a digital signature on $c^\star\|m_i$. Therefore, $(\sigma^\star, c^\star\|m_i)$ will form a valid existential forgery on the underlying digital signature scheme.

C.5 Proof of Theorem 2

Proof. Let $\mathcal{A}$ be $(t, \epsilon, q_s, q_v, q_u, q_{cd}, q_{pv})$-IND-CCA attacker against the above signcryption scheme sc which uses (t, ϵ', q_s)-EUF-CMA secure digital signature and an IND-OT secure DEM $\mathcal{D}$. We build a $(t + q_s(q_u + q_{cd}, q_{pv}), \epsilon \cdot (1-\epsilon')^{q_u+q_{cd}+q_{pv}})$-IND-CPA attacker $\mathcal{R}$ against the underlying KEM as follows.

keygen **and** setup. $\mathcal{R}$ gets the public key $\mathcal{K}.\mathsf{pk}$ of the KEM $\mathcal{K}$ from his challenger. Then, he chooses an appropriate IND-OT secure DEM $\mathcal{D}$ and a (t, ϵ', q_s)-EUF-CMA secure signature scheme Σ along with a key pair $(\Sigma.\mathsf{sk}, \Sigma.\mathsf{pk})$.

signcrypt **and** proveValidity **queries**. On a message m_i, $\mathcal{R}$ will proceed as the standard algorithm with the exception of maintaining in a list, say $\mathcal{L}$, the queried message m_i and the resulting signcryption μ_i, in addition to the intermediate values used to produce this signcryption, i.e. the produced encapsulation c_i, the corresponding decapsulation k_i, the input nonce r_i to the encapsulation algorithm, and finally the digital signature σ_i on $c_i \| m_i$. To verify such a generated signcryption, $\mathcal{R}$ will run the standard proveValidity algorithm. This simulation is clearly indistinguishable from the standard algorithm.

unsigncrypt **queries**. To unsigncrypt $\mu_i = (c_i, e_i)$, $\mathcal{R}$ will look up the list $\mathcal{L}$ for a record containing the encapsulation c_i. If such a record exists, then $\mathcal{R}$ will use the corresponding key, say k_i, to decrypt the query and recover the digital signature and the message m_i then output the last item, otherwise he will output $\perp$.

This simulation departs from the real algorithm when μ_i is a valid signcryption on some message m_i, and $\mathcal{R}$ outputs $\perp$. Two cases, either m_i has never been queried to the signcryption oracle, or not. The first case would correspond to an existential forgery on sc, and thus to an existential forgery on the underlying digital signature scheme by virtue of Theorem 1. The second case would correspond to an existential forgery on the underlying signature scheme according to Remark 8. Hence, the probability that both scenarios do not happen is at least $(1-\epsilon')^{q_u}$ because the underlying digital signature scheme is (t, ϵ', q_s)-EUF-CMA secure by assumption.

$\{$confirm, deny$\}$ **queries**. For a verification query $\mu_i = (c_i, e_i)$ on a message m_i, $\mathcal{R}$ will proceed as above (in the unsigncryption queries); he will look up the list $\mathcal{L}$ for a record containing c_i, if found, $\mathcal{R}$ will prove in ZK the validity of the signcryption of μ_i w.r.t. m_i using the last field of the record, i.e. the input to the encapsulation algorithm used to produce c_i. Otherwise, $\mathcal{R}$ will simulate the deny protocol. This proof is possible to issue since the protocol in question is ZK and thus simulatable using the popular rewinding technique.

This simulation differs from the real algorithm when $\mathcal{R}$ issues the deny protocol when μ_i is a valid signcryption on the message m_i. Two cases, either m_i has not been queried for signcryption in which case (μ_i, m_i) forms an existential forgery on sc, or m_i has been queried before but c_i was never used to generate signcryptions, which corresponds to an existential forgery on the underlying signature scheme according to Remark 8. Both scenarios do not occur with probabilities at least $(1-\epsilon')^{q_{cd}}$.

queries. $\mathcal{R}$ will proceed as in a $\{$confirm, deny$\}$ queries with the exception of issuing the corresponding digital signature instead of the confirm protocol, and $\perp$ instead of the deny protocol. This simulation is correct with probability at least $(1-\epsilon')^{q_{pv}}$.

Challenge. At some point $\mathcal{A}$ will output two messages m_0 and m_1. $\mathcal{R}$ will use his challenge (c, k) to produce a digital signature σ_b on m_b for $b \xleftarrow{R} \{0,1\}$ (chosen by $\mathcal{R}$). Finally, he will produce an encryption e_b on σ_b using k, and outputs $\mu = (c, e_b)$ as a challenge signcryption to $\mathcal{A}$. Note that the challenge (c, k) is computed as follows; if some $b' \xleftarrow{R} \{0,1\}$ is 1, then k is the decapsulation of c, otherwise k is a string chosen uniformly at random from the key space. Thus, if k is the decapsulation of c ($b' = 1$), then μ is a valid signcryption on m_b. Otherwise, it is not a valid signcryption on either messages.

If the advantage of $\mathcal{A}$ is non-negligibly different from the advantage of an indistinguishability adversary in a real attack, i.e. $\mathcal{A}$ responds with $1-b$ with high probability when k is not the decapsulation of c (to denote that μ is not a signcryption of m_b), then $\mathcal{A}$ can be easily turned into an attacker against the IND-OT security property of the DEM underlying the construction. In fact, when $\mathcal{A}$ hands the messages m_0, m_1 to $\mathcal{R}$ (adversary against the IND-OT property of the DEM), this latter produces a pair $(k, c) = \mathcal{K}.\mathsf{encap}()$ consisting of a key and of its encapsulation, and then gives $m_0 \| \Sigma.\mathsf{sign}(m_0 \| c)$ and $m_1 \| \Sigma.\mathsf{sign}(m_1 \| c)$ as challenge messages to his IND-OT challenger. Upon receipt of the challenge encryption e of $m_b \| \Sigma.\mathsf{sign}(m_b \| c)$ for some $b \xleftarrow{R} \{0,1\}$, $\mathcal{R}$ will forward (c, e) to $\mathcal{A}$. When this latter answers with b', then $\mathcal{R}$ will respond with $1-b'$.

To sum up, under the IND-OT assumption of the DEM underlying the construction, the challenge μ is compatible with the indistinguishability game described in 6.

Post challenge phase During this phase, $\mathcal{A}$ continues to issue signcrypt, proveValidity, unsigncrypt, $\{$confirm, deny$\}$, and publicVerify queries, and $\mathcal{R}$ continues to handle them as described previously. Note that in this phase, $\mathcal{A}$ might request the unsigncryption or confirmation/denial of a signcryption which comprise c in its first field; in this case $\mathcal{R}$ will respond with $\perp$ in case of an unsigncryption and will simulate the denial protocol in case of a $\{$confirm, deny$\}$ query.

Again the probability that this simulation is correct is at least $(1-\epsilon')^{q_{cd}+q_u+q_{pv}}$.

Final output When $\mathcal{A}$ outputs his guess b_a, $\mathcal{R}$ will output $b_r = 1$ to his own IND-CPA challenger if $b = b_a$ and 0 otherwise.

The advantage of $\mathcal{A}$ is defined by

$$\epsilon = \left|\Pr(b_a = b|b' = 1) - \frac{1}{2}\right|.$$

$$\begin{aligned}
\mathsf{Adv}(\mathcal{R}) &= (1-\epsilon')^{q_{cd}+q_u}\left|\Pr(b_r = b') - \frac{1}{2}\right| = (1-\epsilon')^{q_{cd}+q_u}\left|\Pr(b_r = 0, b' = 0) + \Pr(b_r = 1, b' = 1) - \frac{1}{2}\right| \\
&= (1-\epsilon')^{q_{pv}+q_{cd}+q_u}\left|\Pr(b_r = 0|b' = 0)\Pr(b' = 0) + \Pr(b_r = 1|b' = 1)\Pr(b' = 1) - \frac{1}{2}\right| \\
&= \frac{(1-\epsilon')^{q_{pv}+q_{cd}+q_u}}{2}\left|\Pr(b_a \neq b|b' = 0) + \Pr(b_a = b|b' = 1) - 1\right| \\
&= \frac{(1-\epsilon')^{q_{pv}+q_{cd}+q_u}}{2}\left|\Pr(b_a = b|b' = 1) - \frac{1}{2}\right| \\
&= \frac{\epsilon(1-\epsilon')^{q_{pv}+q_{cd}+q_u}}{2}
\end{aligned}$$

□

Remark 9. In the theorem above, we considered a statistically IND-OT DEM. In case it is $(t, \epsilon_{\mathcal{D}})$-IND-OT secure, then the advantage of $\mathcal{R}$ will be:

$$\mathsf{Adv}(\mathcal{R}) = \frac{1}{2}(\epsilon + \epsilon_{\mathcal{D}})(1-\epsilon')^{q_{pv}+q_u+q_{cd}}$$

since $\left|\Pr(b_a \neq b|b' = 0) - \frac{1}{2}\right|$ corresponds to the advantage of the IND-OT attacker against the DEM underlying the construction, as explained above.

C.6 Proof of Theorem 3

Proof. Let $\mathcal{A}$ be a (t, ϵ, q_s)-EUF-CMA attacker against the CtEaS construction which uses a (t, ϵ_b) binding commitment. We construct a (t, ϵ, q_s)-EUF-CMA against the construction as follows.

setup **and** keygen $\mathcal{R}$ gets the public key pk_S of the signature scheme from his challenger, then generates further the key pair of the encryption scheme (corresponding to the key pair of the receiver) $(\mathsf{sk}_R, \mathsf{pk}_R)$, and finally hands those entities to $\mathcal{A}$.

signcrypt. On a message m, $\mathcal{A}$ will proceed as the standard algorithm except when it comes to generating the digital signature on the commitment c and the encryption e; he will then solicit his own challenger for a signature on (e, c). It is easy to see that this simulation is indistinguishable from that of the standard algorithm.

Final output. Eventually, $\mathcal{A}$ outputs his forgery $(e^\star, c^\star, \sigma^\star)$ on some message $m^\star$ that was never queried before for signature. By construction, $\sigma^\star$ is a valid digital signature on $(e^\star, c^\star)$. It will form an existential forgery on the digital signature scheme if $(e^\star, c^\star)$ was never queried before by $\mathcal{R}$ for a digital signature. Suppose there exists $1 \leq i \leq q_s$ such that $(e^\star, c^\star) = (e_i, c_i)$ where (e_i, c_i, σ_i) was the output signcryption on the query m_i. Since the encryptions e_i are exactly κ-bit strings by assumption, equality of the strings $(e^\star, c^\star)$ and (e_i, c_i) implies equality of their suffixes (that start at the $(\kappa + 1)$-st position), namely $c^\star$ and c_i. The probability that this case ($c^\star = c_i$ given that $m_i \neq m^\star$) does not occur is at most $(1 - \epsilon_b)^{q_s}$ since the commitment is (t, ϵ_b) binding. Thus, $\mathcal{R}$ returns $(\sigma^\star, e^\star \| c^\star)$ as a valid existential forgery against the digital signature in question.

□

C.7 Proof of Theorem 4

Proof. [setup **and** keygen]. The reduction $\mathcal{R}$ gets the public key pk_E of the encryption scheme it is trying to attack, and generates the parameters for the remaining constituents, namely the key pair $(\mathsf{sk}_S, \mathsf{pk}_S)$ for the signature schemes and the public parameters for the commitment scheme.

[$\mathsf{signcrypt}$ **and** $\mathsf{proveValidity}$ **queries**]. $\mathcal{R}$ proceeds as the standard algorithm with the exception of keeping in a list $\mathcal{L}$ the queried message, the answers, and the intermediate values (random nonces used to commit to the signcrypted message, along with the randomness used to create their encryptions) used to produce the signcryption.

[$\mathsf{unsigncrypt}$ **queries**]. For an unsigncryption query (e_i, c_i, σ_i), $\mathcal{R}$ will output $\perp$ if e_i does not appear in any record of the list $\mathcal{L}$, and the corresponding message otherwise.

This simulation departs from the standard one if $\mathcal{R}$ outputs $\perp$ for a valid query. If the message m_i underlying the query was never asked to the signcryption oracle, then this case would correspond to a valid existential forgery on the signcryption scheme. If m_i was queried before and (e_j, c_j, σ_j) is the corresponding answer ($j < i$), then $(e_j, c_j) \neq (e_i, c_j)$ as $e_i \neq e_j$ and the e_i's are by construction exactly n-bit strings. The query would then lead to a valid existential forgery $(\sigma_i, (e_i, c_j))$ on the underlying signature scheme.

Therefore, the provided simulation is indistinguishable from the execution of the standard algorithm with probability at least $(1 - \epsilon')^{q_u}$.

[$\{\mathsf{confirm}, \mathsf{deny}\}$] For a query (e_i, c_i, σ_i), $\mathcal{R}$ will simulate the denial protocol if e_i does not appear in any record of the list $\mathcal{L}$, and will execute the confirmation protocol otherwise. $\mathcal{R}$ can provide a simulation of deny since the latter is ZK by assumption and thus simulatable. $\mathcal{R}$ can further execute the $\mathsf{confirm}$ protocol thanks to the intermediate values (the nonces used to produce the encryption e_i) kept in the list $\mathcal{L}$.

This simulation deviates from the standard execution of the algorithm with probability at most $(1 - \epsilon')^{q_{cd}}$ for the same reasons explained earlier (in the simulation of unsigncryption queries).

[$\mathsf{publicVerify}$] For a query (e_i, c_i, σ_i), $\mathcal{R}$ will output $\perp$ if e_i does not appear in any record of the list $\mathcal{L}$, and will output the decryption of e_i otherwise.

Again, this simulation deviates from the standard execution of the algorithm with probability at most $(1 - \epsilon')^{q_{pv}}$ for the same reasons explained earlier.

[Challenge phase] Eventually, $\mathcal{A}$ outputs two challenge messages m_0, m_1. $\mathcal{R}$ will choose $b \xleftarrow{R} \{0, 1\}$ and then produce a bit strings r and hands $m_0^\star = (0, 0)$ and $m_1^\star = (m_b, r)$ to his own challenger. He gets as a challenge challenge $e_{b'}$, encryption of $m_{b'}^\star$, for some $b' \xleftarrow{R} \{0, 1\}$. He will then produce a commitment c_b of (m_b, r), and finally generate a signature σ_b on $(e_{b'}, c_b)$. The challenge signcryption output to $\mathcal{A}$ is the triple $\mu = (e_{b'}, c_b, \sigma_b)$.

If $b' = 1$, then $(e_{b'}, c_b, \sigma_b)$ is a valid signcryption of m_b, otherwise it is not a valid signcryption on neither m_b nor m_{1-b}. If $\mathcal{A}$'s advantage is non-negligibly different from that of an indistinguishability adversary in a real attack, i.e. $\mathcal{A}$ answers $1 - b$ with high probability when $b' = 0$, then $\mathcal{A}$ can be turned into an adversary against the hiding property of the underlying commitment scheme as follows. When $\mathcal{A}$ gives the challenge messages m_0, m_1 to $\mathcal{R}$, this latter will pass them to his challenger and will receive a commitment $c_b = \mathsf{commit}(m_b, r)$ on the message m_b for some $b \xleftarrow{R} \{0, 1\}$ and some random nonce r. $\mathcal{R}$ will compute $e_{b'} = \mathsf{encrypt}(0, 0)$. Finally $\mathcal{R}$ produces a digital signature on σ_b on $(e_{b'}, c_b)$, and gives the triple $(e_{b'}, c_b, \sigma_b)$. If $\mathcal{A}$ answers $1 - b$ with high probability, then $\mathcal{A}$ will output b to his own challenger.

To some up, provided the used commitment is hiding, the challenge signcryption $(e_{b'}, c_b, \sigma_b)$ is compatible with the standard indistinguishability game.

[Post challenge phase] $\mathcal{A}$ continue to issue queries to $\mathcal{R}$ who will handle them as previously. Note that from now on, the adversary may ask the unsigncryption/confirmation/denial of $(e_{b'}, c_b, -) \neq \mu$. The probability that this query is invalid is at least $(1 - \epsilon')^{q_{cd} + q_u + q_{pv}}$ since the underlying signature scheme is (t, ϵ', q_s)-SEUF-CMA secure.

[Final output] Eventually, $\mathcal{A}$ outputs a bit b_a. If $b_a = b$, then $\mathcal{R}$ outputs $b_r = 1$ to his challenger, otherwise he answers $b_r = 0$.

The advantage of $\mathcal{A}$ is given by

$$\epsilon = \Pr[b_a = b | b' = 1].$$

Similarly, the advantage of $\mathcal{R}$ is computed as:

$$\begin{aligned}
\mathsf{Adv}(\mathcal{R}) &= (1-\epsilon')^{q_{cd}+q_u+q_{pv}} \left|\Pr(b_r = b') - \frac{1}{2}\right| = (1-\epsilon')^{q_{cd}+q_u+q_{pv}} \left|\Pr(b_r = 0, b' = 0) + \Pr(b_r = 1, b' = 1) - \frac{1}{2}\right| \\
&= (1-\epsilon')^{q_{cd}+q_u+q_{pv}} \left|\Pr(b_r = 0 | b' = 0)\Pr(b' = 0) + \Pr(b_r = 1 | b' = 1)\Pr(b' = 1) - \frac{1}{2}\right| \\
&= \frac{(1-\epsilon')^{q_{cd}+q_u+q_{pv}}}{2} \left|\Pr(b_a \neq b | b' = 0) - \frac{1}{2} + \Pr(b_a = b | b' = 1) - \frac{1}{2}\right| \\
&= \frac{(1-\epsilon')^{q_{cd}+q_u+q_{pv}}}{2} |\epsilon_h + \epsilon| \\
&= \frac{(\epsilon+\epsilon_h)(1-\epsilon')^{q_{cd}+q_u+q_{pv}}}{2}
\end{aligned}$$

□

D Efficient Verifiable Signcryption from the EtS Paradigm

D.1 Proof of Theorem 5

Proof. The adversary $\mathcal{R}$ against the signature underlying the construction will get the parameters of the target digital signature from his challenger. Then, he generates the parameters for the encryption scheme. For a signcryption query on message m_i, $\mathcal{R}$ computes an encryption e_i of m_i, then requests his challenger for a signature on e_i. Let σ_i be the answer of such a query. $\mathcal{R}$ will then output (e_i, σ_i).

Eventually, the adversary $\mathcal{A}$ against the signcryption construction will output a forgery $(e^\star, \sigma^\star)$ on a message $m^\star$, that was never queried before. $\sigma^\star$ is by definition a digital signature on $e^\star$. The last item was never queried by $\mathcal{R}$ for digital signature, since otherwise $m^\star$ would have been queried before. We conclude that $(e^\star, \sigma^\star)$ is a valid forgery on the digital signature scheme. □

D.2 Proof of Theorem 6

Proof. The IND-CPA adversary $\mathcal{R}$ against the encryption scheme gets the parameters from his challenger, and generates the parameters for a (t, ϵ', q_s)-SEUF-CMA secure digital signature . For a signcryption query on m_i, $\mathcal{R}$ will proceed as in the real algorithm, with the exception of maintaining a list $\mathcal{L}$ of records that consists of the query, its encryption, the randomness used to produce the encryption, and finally the digital signature on the encryption. For a verification query, $\mathcal{R}$ will proceed as in the standard proveValidity protocol . And finally, for an unsigncryption, a confirmation/denial, or a public verification query (e_i, σ_i), $\mathcal{R}$ will look up the list $\mathcal{L}$ (after checking the validity of the signature σ_i on the message m_i); if e_i appears in the list as a used encryption, then $\mathcal{R}$ will answer with the corresponding message in case of an unsigncryption query, will issue a NIZK proof of the correctness of the decryption (using the randomness used to generate e_i) in case of a public verification query, and will execute the confirm protocol (using the randomness applied to generate the encryption) in case of a confirmation/denial query. Otherwise, $\mathcal{R}$ will issue $\perp$ in case of an unsigncryption or public verification query, and will simulate the deny protocol in case of a confirmation/denial query.
This simulation departs from the real one when the queried signcryption (e_i, σ_i) is valid on m_i however e_i is not in the list $\mathcal{L}$. We distinguish two cases, either the message in question m_i was not queried before for signcryption, in which case such a query would correspond to a valid existential forgery on the construction, and thus on the underling signature scheme. Or, the queried signcryption is on a message that has been queried before, which corresponds to an existential forgery on the underlying signature scheme. Since the signature scheme underlying the construction is (t, ϵ', q_s)-SEUF-CMA secure, this scenario does not happen with probability at least $(1-\epsilon')^{q_u+q_{cd}+q_{pv}}$.

Eventually, $\mathcal{A}$ produces two messages m_0, m_1. $\mathcal{R}$ will forward the same messages to his challenger and obtain a ciphertext e, encryption of m_b for some $b \xleftarrow{R} \{0,1\}$. $\mathcal{R}$ will produce a digital signature σ on e and give the result in addition to e to $\mathcal{A}$ as a challenge signcryption. It is easy to see that $\mathcal{A}$'s answer is sufficient for $\mathcal{R}$ to conclude. Note that after the challenge phase, $\mathcal{A}$ is allowed to make the previous queries and $\mathcal{R}$ can handle them as previously. There is however the possibility for $\mathcal{A}$ of issuing an unsigncryption query of the type $(c, -) \neq (c, \sigma)$ (confirmation/denial or public verification query on $(c, -) \neq (c, \sigma)$ and m_b). $\mathcal{R}$ will respond to such a query by issuing the $\perp$ symbol (denial protocol). The probability that this answer does not differ from the output of the real algorithm is at least $(1-\epsilon')^{q_u+q_{cd}+q_{pv}}$ as the signature scheme underlying the construction is (t, ϵ', q_s)-SEUF-CMA secure by assumption. $\square$

D.3 Proof of Theorem 9

Proof. The adversary $\mathcal{R}$ against the signature underlying the construction will proceed as the real algorithm with the exception of requesting his challenger when it comes to computing the digital signature on the ciphertexts.

Eventually, the unforgeability adversary $\mathcal{A}$ against the signcryption construction will output a forgery $(c^\star, \sigma^\star)$ on a message $m^\star$, that was never queried before for signcryption w.r.t. the target receiver's key $\mathsf{pk}_R^\star$. Since $\sigma^\star$ is by definition a digital signature on $c^\star \| pk_R^\star$, then the latter item was never queried by $\mathcal{R}$ for digital signature. We conclude that $(c^\star \| pk_R^\star, \sigma^\star)$ is a valid forgery on the digital signature scheme. $\square$

D.4 Proof of Theorem 10

Proof. Let $\mathcal{R}$ be the IND-sTag-CCA attacker against the tag-based encryption scheme. Once $\mathcal{A}$ generates the target sender's key pair, namely $(sk_S^\star, \mathsf{pk}_S^\star)$, he forwards $\mathsf{pk}_S^\star$ to his challenger as the selective tag committed to. He gets as a response the public key $\mathsf{pk}_R^\star$ (of the encryption scheme) which he sets as the target receiver's public key.

Queries w.r.t. the sender's key $\mathsf{pk}_S^\star$ are answered as in 6. The probability that such a simulation differs from the standard execution of the algorithms/protocols is at least $(1-\epsilon')^{q_u+q_{cd}+q_{pv}}$.

Signcryption and verification queries w.r.t. a sender's key $\mathsf{pk}_S \neq \mathsf{pk}_S^\star$ are answered as in the standard algorithm/protocol. The remaining queries are perfectly handled using $\mathcal{R}$'s challenger since the tag in question, i.e. pk_S, is different from $\mathsf{pk}_S^\star$. Such a simulation is clearly indistinguishable from the standard execution of the algorithm/protocols in question.

The challenge phase is similar to that in the proof of Theorem 6, and finally the post-challenge phase is simulated like the pre-challenge one.

The response of $\mathcal{A}$ to the challenge is sufficient for $\mathcal{R}$ to conclude. $\square$

D.5 Instantiation with Kitlz' tag-based encryption

Setup	Choose a bilinear group $(\mathbb{G}, \cdot)$ generated by g with prime order d.
Key generation	Choose $x_1, x_2, x_3, x_4 \xleftarrow{R} \mathbb{Z}_d$ then compute $X_i \leftarrow g^{x_i}$, for $1 \leq i \leq 4$,
Encryption	For a message $m \in \mathbb{G}$ and a tag $t \in \mathbb{Z}_d$:
	choose $r_1, r_2 \xleftarrow{R} \mathbb{Z}_d$,
	set the ciphertext to $(Y_1, Y_2, Y_3, Y_4, Y_5) \leftarrow (X_1^{r_1}, X_2^{r_2}, (g^t X_3)^{r_1}, (g^t X_4)^{r_2}, m \cdot g^{r_1+r_2})$.
Decryption	Given a ciphertext $(Y_1, Y_2, Y_3, Y_4, Y_5)$ and a tag t:
	check that $(Y_3, Y_4) = (Y_1^{(t+x_3)/x_1}, Y_2^{(t+x_4)/x_2})$
	if so then output $m = Y_5/(Y_1^{x_1^{-1}} Y_2^{x_2^{-1}})$, otherwise abort

Fig. 3. Kiltz' tag-based encryption scheme

Fact 2 *Let c be an encryption of some message m using Kiltz' tag-based encryption scheme under tag t w.r.t. some public key* pk. *Let further m' be an arbitrary message different from m . There exists efficient proofs for:*

- *proving knowledge that m is the decryption of c w.r.t.* pk *under tag t, where the private input of the prover is either the private key corresponding to* pk*, or the randomness used to create c.*
- *proving that m' is not the decryption of c w.r.t.* pk *under tag t, where the private input of the prover is the private key corresponding to* pk*.*
- *proving knowledge of the decryption of c w.r.t.* pk *under tag t, namely the message m, where the private input of the prover is the randomness used to create c.*

Proof. We parse c as $(Y_1, Y_2, Y_3, Y_4, Y_5)$. Note that checking the consistency of the ciphertext c can be publicly achieved. In fact, as we work in a bilinear group, anyone can check the equality of the discrete logarithms of Y_i in base X_i, and of Y_j in base $g^t X_j$, where $(i,j) \in \{(1,3,(2,4)\}$.

- To prove that m is the decryption of c, one needs to prove the equality of the discrete logarithm of X_2 in base g and of $e(X_1, Y_2)$ in base $e(Y_5 m^{-1}, X_1)e(g, Y_1)^{-1}$, where e is the pairing underlying the group $\mathbb{G}$. We refer to [15] for the proof of equality of two discrete logarithms. It is obvious that the private input of the prover in such a proof is the private key of the scheme. If the prover holds only the randomness used to create c, then proving that m is the decryption of c comes to a proof of discrete logarithm formula [11], i.e. the discrete logarithm of $Y_5 m^{-1}$ in base g is the sum of the discrete logarithms of Y_i in base X_i, $i \in \{1, 2\}$.
- To prove that m' isn't the decryption of c, one needs to prove the inequality of of the discrete logarithm of X_2 in base g and of $e(X_1, Y_2)$ in base $e(Y_5 m^{-1}, X_1)e(g, Y_1)^{-1}$ (e is always the pairing underlying $\mathbb{G}$). See [14] for details of such a proof.
- Finally, we provide the proof of knowledge of the decryption of c in Figure 4. This proof departs from that described in Figure 1 only in the use of tags.

Prover P		Verifier V
Choose $m' \xleftarrow{R} \mathcal{G}$		
Compute $c' = \mathsf{Kiltz.encrypt}_{\mathsf{Kiltz.pk}}(\mathsf{m'}, \mathsf{t})$		
$= (X_1^{s_1}, X_2^{s_2}, (g^t X_3)^{s_1}, (g^t X_4)^{s_2}, m \cdot g^{s_1+s_2})$	$\xrightarrow{\quad c' \quad}$	
	$\xleftarrow{\quad b \quad}$	Choose $b \xleftarrow{R} \{0,1\}^\ell$ $(b \in \mathbb{N})$
	$\xrightarrow{\quad z = m' \cdot m^b \quad}$	
	$\mathsf{PoK}\{c' \cdot c^b = \mathsf{Kiltz.encrypt}_{\mathsf{Kiltz.pk}}(\mathsf{z}, \mathsf{t})\}$ $\longleftrightarrow$	
		Accept if PoK is valid

Fig. 4. Proof system for membership to the language $\{m : c = \mathsf{Kiltz.encrypt}_{\mathsf{Kiltz.pk}}(\mathsf{m}, \mathsf{t})\}$ Common input: $(c, \mathsf{Kiltz.pk})$ and Private input: m and randomness used to produce e.

□

E The EtStE Paradigm

E.1 Proof of Theorem 11

Proof. Let $\mathcal{A}$ be a (t, ϵ, q_s)-EUF-CMA adversary against the above construction. A (t, ϵ, q_s)-EUF-CMA adversary $\mathcal{R}$ against the underlying signature scheme, say Σ, proceeds as follows.

After $\mathcal{R}$ gets the parameters of Σ from his challenger, he will choose two suitable encryption schemes conforming with the specifications of the construction, and will hand the key pairs of these schemes along with the public key of Σ to $\mathcal{A}$. Signcyption queries made by $\mathcal{A}$ are answered using $\mathcal{R}$'s challenger (when it comes to producing the digital signature on encapsulation and the encryption of the message in question). Eventually $\mathcal{A}$ outputs his forgery $\mu^\star = (\mu_1^\star, \mu_2^\star, \mu_3^\star, \mu_4^\star)$ on a message $m^\star$. Let $s^\star$ be the decryption of $(\mu_2^\star, \mu_3^\star)$; by construction, $(s^\star, \mu_4^\star)$ is a valid digital signature on $\mu_2^\star \| \mu_1^\star$. Since $m^\star$ was never queried before, then $\mathcal{R}$ never requested his challenger for a digital signature on a string whose suffix (starting at the (κ+1) position) is $\mu_1^\star$. Thus $(s^\star, \mu_4^\star)$ along with $\mu_2^\star \| \mu_1^\star$ forms a valid EUF-CMA forgery on Σ. □

Remark 10. Remark 8 applies also here for this paradigm. In fact, suppose that an adversary $\mathcal{A}$ is able to produce a valid signcryption $\mu^\star = (\mu_1^\star, \mu_2^\star, \mu_3^\star, \mu_4^\star)$ on a message m which has been queried before, but where the encapsulation $\mu_2^\star$ was never used to answer $\mathcal{A}$'s signcryption queries. Let further $s^\star$ be the decryption of $(\mu_2^\star, \mu_3^\star)$. Then $(s^\star, \mu_4^\star)$ along with $\mu_2^\star \| \mu_1^\star$ forms a valid EUF-CMA forgery on Σ as $\mathcal{R}$ never invoked his challenger for a digital signature on a string whose κ-prefix is $\mu_1^\star$.

E.2 Proof of Theorem 12

Proof. The IND-CPA adversary $\mathcal{R}$ against the KEM gets the public key of the KEM from his challenger and generates key pairs of the remaining constituents that comply with the specifications of the construction. He then hands the public parameters/keys to $\mathcal{A}$, the IND-CCA adversary against the signcryption scheme.

For signcryption/public verification queries, $\mathcal{R}$ will proceed as the standard algorithm with the exception of keeping in a list $\mathcal{L}$ the used encapsulations, the corresponding keys, and the random coins used to produce these quantities. Verification queries are answered as usual using the intermediate values used to create the signcryption. For unsigncryption queries $(\mu_1, \mu_2, \mu_3, \mu_4)$, $\mathcal{R}$ will look up $\mathcal{L}$ for μ_2, if it is found in the list, then he proceeds as the standard algorithm using the decapsulation of μ_1, otherwise he outputs $\perp$. Similarly, for confirm/deny queries, $\mathcal{R}$ will look up $\mathcal{L}$, if the second component of the presumed signcryption appears in the list, then $\mathcal{R}$ proceeds as the standard protocol using the corresponding key, otherwise he will simulate the deny algorithm.

The provided simulation deviates from the standard execution of the indistinguishability game when $\mathcal{R}$ outputs $\perp$ or simulates the deny algorithm for a valid signcryption μ. Two cases manifest, either the message in question has not been queried for signcryption in which case μ forms a valid forgery on the construction, or the message has been queried but the encapsulation (second component of μ) was never used to answer signcryption queries. According to Remark 10, this case corresponds to an EUF-CMA forgery on the underlying signature scheme. Thus, the above simulation is indistinguishable from the standard execution of the indistinguishability game with probability at least $(1-\epsilon')^{q_u+q_{cd}+q_{pv}}$ since the used signature scheme is (t, ϵ', q_s)-EUF-CMA.

Eventually, $\mathcal{A}$ outputs two challenge messages m_0, m_1. $\mathcal{R}$ will encrypt, in e_b, the message m_b, for $b \xleftarrow{R} \{0,1\}$. Next, he produces a signature (r, s) on $c \| e_b$, where (c, k) is his challenge. Finally, $\mathcal{R}$ encrypts s in $e_{\mathcal{D}}$ using k, and outputs $\mu = (e_b, c, e_{\mathcal{D}}, r)$ as a challenge signcryption. Since the used encryption is IND-CPA secure by assumption, then information about m_b can be only leaked from $(c, e_{\mathcal{D}}, r)$. If k is the decapsulation of c, then μ is a valid signcryption of m_b, otherwise it is not a valid signcryption on either messages. The rest of the proof follows as in that of Theorem 2. Note that after the challenge phase, $\mathcal{A}$ can ask for unsigncryption, confirmation/denial, and public verification any signcryption that is different from μ; the strong unforgeability of the underlying signature scheme is not needed in this phase because $\mathcal{A}$ does not have in clear the signature (r, s). As a result, $\mathcal{R}$ will break the IND-CPA security of the KEM underlying the construction with probability at least $\frac{\epsilon(1-\epsilon')^{q_{cd}+q_u+q_{pv}}}{2}$. □

E.3 The class $\mathbb{S}$ of used signature schemes

Definition 9. *$\mathbb{S}$ is the set of all digital signatures for which there exists a pair of efficient 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. *r reveals no information about m nor about* pk*, i.e. there exists an algorithm* simulate *such that for every public key* pk *from the key space and for every message m from the message space, the output* simulate(pk, m) *is statistically indistinguishable from r.*
2. *there exists an algorithm* compute *that on the input* pk*, the message m and r, computes a description of an injective* one-way function $f : (\mathbb{G}, *) \to (\mathbb{H}, \circ_s)$*:*
   - *where $(\mathbb{G}, *)$ is a 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 $\mathsf{retrieve}$ *is an algorithm that inputs* pk, m *and the correctly converted pair* (s, r) *and retrieves the signature* σ *on* m.

This class encompasses most signature schemes described in the literature, e.g. [5, 44, 26, 9, 42, 19, 12, 13, 7, 50, 49].

Proving knowledge of the decryption of the ciphertext $(c, e_{\mathcal{D}})$ (produced using $(\mathcal{K}, \mathcal{D})$), and that this decryption forms, along with some r, a valid/invalid digital signature on a known string, say $c \| e$, can be accomplished as follows:

Receiver R		Verifier V
Compute I as defined in Definition 9		Compute I as defined in Definition 9
Choose $s' \xleftarrow{R} \mathbb{G}$		
Compute $t_1 = f(s')$		
Compute $t_2 = \{\mathcal{K}, \mathcal{D}\}.\mathsf{encrypt}_{\mathcal{K}.\mathsf{pk}}(s')$	$\xrightarrow{t_1, t_2}$	
	$\xleftarrow{b}$	Choose $b \xleftarrow{R} \{0,1\}^{\ell}$ $(b \in \mathbb{N})$
	$\xrightarrow{z = s' * s^b}$	
	$\mathsf{PoK}\{z = \{\mathcal{K}, \mathcal{D}\}.\mathsf{decrypt}_{\mathcal{K}.\mathsf{sk}}(t_2 \circ_e (e, s_k)^b)\}$ $\longleftrightarrow$	
		Accept if the proof PoK is valid and, $f(z) = t_1 \circ_s I^b$ in case of confirmation, $f(z) \neq t_1 \circ_s I^b$ in case of denial.

Fig. 5. Proof system for membership to the language $\{(c, e_{\mathcal{D}}) \colon \exists s \ : \ s = \{\mathcal{K}, \mathcal{D}\}.\mathsf{decrypt}(c, e_{\mathcal{D}}) \wedge \Sigma.\mathsf{proveValidity}(\mathsf{retrieve}(s, r), c \| e) = (\neq) 1\}$ Common input: $(e, c, e_{\mathcal{D}}, r, \Sigma.\mathsf{pk}, \mathcal{K}.\mathsf{pk})$ and Private input: $\mathcal{K}.\mathsf{sk}$ or randomness encrypting s in $(c, e_{\mathcal{D}})$