Confidential Signatures and Deterministic Signcryption

Alexander W. Dent[1], Marc Fischlin[2], Mark Manulis[2], Martijn Stam[3], and Dominique Schröder[2]

[1] Royal Holloway, University of London, U.K.
[2] Darmstadt University of Technology, Germany
[3] LACAL, EPFL, Switzerland

Abstract. Encrypt-and-sign, where one encrypts and signs a message in parallel, is usually not recommended for confidential message transmission. The reason is that the signature typically leaks information about the message. This motivates our investigation of confidential signature schemes, which hide all information about (high-entropy) input messages. In this work we provide a formal treatment of confidentiality for such schemes and a comprehensive discussion of the relationship of different notions we propose. We give constructions meeting our notions, both in the random oracle model and the standard model. As part of this we show that full domain hash signatures achieve a weaker level of confidentiality than Fiat-Shamir signatures. We then revisit the connection of confidential signatures to signcryption schemes. We give formal security models for deterministic signcryption schemes for high-entropy and low-entropy messages, and prove encrypt-and-sign to be secure for confidential signature schemes and high-entropy messages. Finally, we show that one can derandomize any signcryption scheme in our model and obtain a secure deterministic scheme.

1 Introduction

A common mistake amongst novice cryptographers is to assume that digital signature schemes provide some kind of confidentiality service to the message being signed. The (faulty) argument in support of this statement is (a) that all signature schemes are of the "hash-and-sign" variety, which apply a hash function to a message before applying any kind of keyed operation, and (b) that a one-way hash function will hide all partial information about a message. Both facets of this argument are incorrect. However, it does suggest that notions of confidentiality for signature schemes are an interesting avenue of research.

The question of confidentiality of hash functions in signature schemes was previously considered by Canetti [7] as "content-concealing signatures"; however the original treatment is naïve and serves only to motivate the concept of perfect one-way hash functions [7, 8]. We provide a more formal treatment here. The question of entropic security has also been consider by several other authors. Dodis and Smith studied entropic secure primitives requiring that no function leaks whatsoever their input [11]. Russell and Wang [20] consider the security of symmetric encryption schemes based on high-entropy messages; whereas several authors have considered the security of asymmetric encryption schemes based on high-entropy messages [3, 4, 6]. However, we are the first authors to consider the confidentiality of signatures and signcryption schemes with respect to high entropy messages.

Defining Confidential Signatures. Our first contribution is to define confidential signatures. Our starting point are high-entropy messages (signatures for messages with low entropy inevitably leak through the verification algorithm of the signature scheme). Our definitions are based on previous efforts for highly-entropic, deterministic public-key encryption [3], and yield three versions of confidential signature schemes:

- Weak confidentiality means that no information is leaked to a passive adversary, except possibly for information related to the technical details of the signature scheme.
- Mezzo confidentiality means that no information is leaked to a passive adversary (in possession of the verification key). Note that this is in contrast to deterministic public-key encryption where information cannot be hidden in such circumstances [3].

– Strong confidentiality means that no information is leaked to an active adversary.

Our definition is general enough to cover probabilistic and deterministic signature schemes, although we need an additional stipulation in the latter case, preventing the case where the information to be leaked is the unique signature itself.

For deterministic public-key encryption several definitions have now been investigated and their equivalence has been shown by Bellare *et al.* [4]. Here we adapt their notions and introduce corresponding definitions for confidential signatures. We also discuss the relationship among the notions, showing that our original notion is robust with respect to such variations.

Constructing Confidential Signatures. We then show how to obtain confidential signatures. To this end, we first introduce the related concept of confidential hash functions, akin to hiding hash functions [3]. We prove that random oracles are confidential hash functions, as are perfectly one-way hash functions [7, 8] in a weaker form.

We then show that the use of weakly confidential hash functions in full domain hash (FDH) signature schemes yields weakly confidential signatures. In particular, this means that FDH signature schemes are weakly confidential in the random oracle model. We show that Fiat-Shamir signatures are strongly confidential in the random oracle model. We also show that strongly secure confidential signatures can be obtained in the standard model via the use of a randomness extractor [18, 17] (provided the entropy of the messages lie above some fixed bound).

Applications to Signcryption. Secure message transmission is usually performed via the encrypt-then-sign paradigm, where the sender encrypts the message under the receiver's public encryption key and then signs the ciphertext with his own signing key. The idea behind signcryption schemes, introduced by [21], aims to gain efficiency by somehow combining the two operations. One consequence of previous security definitions for signcryption schemes [1, 2] is that the encrypt-and-sign approach, where one encrypts the message and signs the message in parallel, does not provide a secure signcryption in general as the signature may reveal information about the message.

We introduce security notions for (possibly deterministic) signcryption schemes with high-entropy messages, along the lines of deterministic public-key encryption and confidential signatures. In case of signcryption schemes we can also give a low-entropy version (explicitly prohibiting the adversary to signcrypt either challenge message) and show that this definition is strictly stronger than the definitions for high-entropy messages. We show that the parallelizable encrypt-and-sign is high-entropy confidential if the underlying encryption scheme is IND-CCA2 and the signature scheme is confidential (and deterministic). We finally prove that we can derandomize any signcryption scheme to derive a secure deterministic scheme.

2 Confidential Signature Schemes

We formalise the notion of a confidential signature in three ways and give constructions. These confidentiality notions can be applied to either probabilistic or deterministic signature schemes.

2.1 Definition of Confidential Signature Schemes

A digital signature scheme is a tuple of efficient algorithms $\mathsf{SS} = (\texttt{SS.Setup}, \texttt{SS.Kg}, \texttt{SS.Sign}, \texttt{SS.Ver})$. The parameter generation algorithm produces a set of parameters common to all users $\lambda_{ss} \xleftarrow{R} \texttt{SS.Setup}(1^k)$, subsequently the key generation algorithm produces a public/private key pair $(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$. The signing algorithm takes a message $m \in \{0,1\}^*$ and the private key, and outputs a signature $\sigma \xleftarrow{R} \texttt{SS.Sign}(sk, m)$. The verification algorithm takes as input a message, signature and public key, and outputs either a valid symbol $\top$ or an invalid symbol $\perp$. This is written $\texttt{SS.Ver}(pk, m, \sigma)$. The standard notion for signature security is that of unforgeability under chosen message attacks (see Appendix A.1 for formal definitions).

We present three confidentiality notions for a digital signature scheme — see Figure 1. These notions are split depending on the adversaries capabilities, which corresponds in a natural way to

$Expt_{\mathcal{A}}^{wSig-b}(k)$:
$\lambda_{ss} \xleftarrow{R} \texttt{SS.Setup}(1^k)$
$(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$
$(\boldsymbol{m}_0, t_0) \xleftarrow{R} \mathcal{A}_1(1^k)$
$(\boldsymbol{m}_1, t_1) \xleftarrow{R} \mathcal{A}_1(1^k)$
$\boldsymbol{\sigma}^* \leftarrow \texttt{SS.Sign}(sk, \boldsymbol{m}_b)$
$t' \xleftarrow{R} \mathcal{A}_2^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk, \boldsymbol{\sigma}^*)$
If $t' = t_0$ then output 1
Else return 0

$Expt_{\mathcal{A}}^{mSig-b}(k)$:
$\lambda_{ss} \xleftarrow{R} \texttt{SS.Setup}(1^k)$
$(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$
$(\boldsymbol{m}_0, t_0) \xleftarrow{R} \mathcal{A}_1(1^k, pk)$
$(\boldsymbol{m}_1, t_1) \xleftarrow{R} \mathcal{A}_1(1^k, pk)$
$\boldsymbol{\sigma}^* \leftarrow \texttt{SS.Sign}(sk, \boldsymbol{m}_b)$
$t' \xleftarrow{R} \mathcal{A}_2^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk, \boldsymbol{\sigma}^*)$
If $t' = t_0$ then output 1
Else return 0

$Expt_{\mathcal{A}}^{sSig-b}(k)$:
$\lambda_{ss} \xleftarrow{R} \texttt{SS.Setup}(1^k)$
$(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$
$(\boldsymbol{m}_0, t_0) \xleftarrow{R} \mathcal{A}_1^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk)$
$(\boldsymbol{m}_1, t_1) \xleftarrow{R} \mathcal{A}_1^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk)$
$\boldsymbol{\sigma}^* \leftarrow \texttt{SS.Sign}(sk, \boldsymbol{m}_b)$
$t' \xleftarrow{R} \mathcal{A}_2^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk, \boldsymbol{\sigma}^*)$
If $t' = t_0$ then output 1
Else return 0

Fig. 1. Notions of confidentiality for (a) weakly confidential signature schemes; (b) mezzo confidential signature schemes; (c) strongly confidential signature schemes

real-life scenarios where it may be possible to derive some information about a message from a signature which might be deemed practically useless, e.g., the value of the hash of the message, but leakage of which cannot be avoided.

In the weak confidentiality notion, the attacker should not be able to determine any information about the messages apart from that which can be obtained directly from the signature itself. The mezzo confidentiality notion models the scenario where the attacker is able to retrieve public-key of the users (before attempting to influence their behaviour), but cannot interact directly with their communication network and obtain signatures of messages. In the strong notion of security, the attacker should not be able to determine any information about the messages apart from the signature itself.

For $x \in \{w, m, s\}$, the attacker $\mathcal{A}$'s advantage in the $xSig$ game is defined to be:

$$Adv_{\mathcal{A}}^{xSig}(k) = |\Pr[Expt_{\mathcal{A}}^{xSig-0}(k) = 1] - \Pr[Expt_{\mathcal{A}}^{xSig-1}(k) = 1]|\,.$$

A signature scheme is weakly confidential (resp. mezzo confidential/strongly confidential) if all PPT attackers $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ have negligible advantage $Adv_{\mathcal{A}}^{xSig}(k)$ in the $wSig$ (resp. $mSig/sSig$) security game, subject to the following restraints:

- Pattern preserving: there exists a length function $\ell(k)$ and equality functions $\diamond_{ij} \in \{=, \neq\}$ $(1 \leq i, j \leq \ell(k))$ such that for any admissible input a in the corresponding game and all possible $(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(a)$ we have that $|\boldsymbol{m}| = \ell(k)$ and $m_i \diamond_{ij} m_j$.
- High entropy: the function $\mu(k) = \max_{m \in \{0,1\}^*} \Pr[m_i = m : (\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(a)]$ is negligible for all $i \in \mathbb{N}$ and for any admissible input a in the corresponding game. This is termed the adversary's *min entropy.*

For deterministic schemes we need the following additional constraint, ruling out trivial attacks:

- Signature free: $\mathcal{A}_1$ does not output a message $m_i \in \boldsymbol{m}$ where it has queried the signature oracle on m_i. (This security requirement only affects strongly confidential signature schemes.)

The latter condition prevents an attacker against a deterministic scheme from "winning" by setting $t \leftarrow \mathcal{S}(sk, m)$ – i.e., it prevents the attacker from "winning" the game simply by determining that the message m has the property that its unique signature is $\mathcal{S}(sk, m)$.

The notions of confidentiality are strictly increasing in strength. If SS is a weakly confidential signature schemes, then Figure 2 is a scheme which is weakly confidential but not mezzo confidential. Similarly, if SS is a mezzo confidential signature scheme, then Figure 3 is a scheme which is mezzo confidential but not strongly confidential. The appropriate proofs of security for these constructions are given in Appendix B.

2.2 Relation to Other Notions of Confidentiality

In this section, we investigate the relationship between the notion of confidentiality that we have proposed to other possible notions of confidentiality in a manner similar to Bellare *et al.* [4]. We

```
SS.Kg′(λ_ss):
  r ←R {0,1}^k
  (pk, sk) ←R SS.Kg(λ_ss)
  Return (pk‖r, sk‖r)

SS.Sign′(sk‖r, m):
  If m = m′‖r
    Return SS.Sign(sk, m)‖m
  Else
    Return SS.Sign(sk, m)

SS.Ver′(pk‖r, m, σ):
  If m = m′‖r
    Parse σ as σ′‖m
    σ ← σ′
  Return SS.Ver(pk, m, σ)
```

Fig. 2. A signature scheme which is weakly confidential but not mezzo confidential

```
SS.Kg′(λ_ss):
  r ←R {0,1}^k
  (pk, sk) ←R SS.Kg(λ_ss)
  Return (pk, sk‖r)

SS.Sign′(sk‖r, m):
  Return SS.Sign(sk, m‖r)‖r

SS.Ver′(pk, m, σ):
  Parse σ as σ′‖z for |z| = k
  Return SS.Ver(pk, m‖z, σ)
```

Fig. 3. A signature scheme which is mezzo confidential but not strongly confidential

define simulator-based notions of security $xSig'$ for $x \in \{w, m, s\}$ and boolean/balanced versions of both the computational and simulation-based security notions. We give relations between these notions which show that these notions are equivalent.

We start by defining simulation-based security notions. These are given in Figure 4. We define an attacker/simulator advantage to be

$$Adv^{xSig'}_{\mathcal{A},S}(k) = |\Pr[Expt^{xSig'-1}_{\mathcal{A},S}(k) = 1] - \Pr[Expt^{xSig'-0}_{\mathcal{A},S}(k) = 1]|$$

and a scheme is declared to be $xSig'$ secure if for all PPT attackers $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ there exists a PPT simulator S such that $Adv^{xSig'}_{\mathcal{A},S}(k)$ is negligible (subject to the restriction that $\mathcal{A}$ is pattern preserving, high entropy, and possibly signature free).

We define a scheme to be boolean $xSig$-secure (resp. boolean $xSig'$-secure) if it is $xSig$-secure (resp. $xSig'$-secure) for PPT attackers $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ where $\mathcal{A}_1(a)$ outputs $(\boldsymbol{m}, t)$ with $|t| = 1$. A scheme is δ-balanced $xSig$-secure (resp. δ-balanced $xSig'$-secure) if it is $xSig$-secure (resp. $xSig'$-secure) for PPT attackers $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ with $\mathcal{A}_1(a)$ outputs $(\boldsymbol{m}, t)$ where $|t| = 1$ and

$$|\Pr[t = b] - 1/2| \leq \delta \text{ for any } b \in \{0, 1\}\,.$$

It is clear (by inspection) that a scheme which is $xSig'$ secure is necessarily boolean $xSig'$-secure and that a scheme which is boolean $xSig'$-secure is necessarily δ-balanced $xSig'$-secure for any value of δ.

Proposition 1. *We establish equivalence using the following results:*

1. *A scheme is δ-balanced xSig secure if it is δ-balanced $xSig'$ secure for some negligible value $\delta(k)$ where $x \in \{w, m, s\}$.*
2. *A scheme is δ-balanced xSig secure for any fixed value of δ if it is 0-balanced xSig secure where $x \in \{w, m, s\}$.*
3. *A scheme is boolean xSig secure if it is δ-balanced xSig for some $\delta(k) \geq 1/p(k)$ where $x \in \{w, m, s\}$ and $p(k)$ is some positive polynomial.*
4. *A scheme is xSig secure if it is boolean xSig where $x \in \{w, m, s\}$.*
5. *A scheme is $xSig'$ secure if it is xSig secure where $x \in \{w, m, s\}$.*

This proposition is proven in Appendix C.

3 Confidential Hash Functions and Signature Schemes

3.1 Confidential Hash Functions

We recap the notion of a *hiding* hash function by Bellare *et al.* [3], but call such functions confidential here. For our purposes, a hash function $\mathsf{H} = (\mathtt{H.Kg}, \mathtt{H})$ is a pair of algorithms for key

$Expt^{wSig'-b}_{\mathcal{A},S}(k)$:
  $\lambda_{ss} \xleftarrow{R} \texttt{SS.Setup}(1^k)$
  $(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$
  $(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(1^k)$
  $\boldsymbol{\sigma}^* \leftarrow \texttt{SS.Sign}(sk, \boldsymbol{m})$
  If $b = 0$ then
    $t' \xleftarrow{R} \mathcal{A}_2^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk, \boldsymbol{\sigma}^*)$
  Else
    $t' \xleftarrow{R} S^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk)$
  If $t' = t$ then output 1
  Else return 0

$Expt^{mSig'-b}_{\mathcal{A},S}(k)$:
  $\lambda_{ss} \xleftarrow{R} \texttt{SS.Setup}(1^k)$
  $(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$
  $(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(1^k, pk)$
  $\boldsymbol{\sigma}^* \leftarrow \texttt{SS.Sign}(sk, \boldsymbol{m})$
  If $b = 0$ then
    $t' \xleftarrow{R} \mathcal{A}_2^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk, \boldsymbol{\sigma}^*)$
  Else
    $t' \xleftarrow{R} S^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk)$
  If $t' = t$ then output 1
  Else return 0

$Expt^{sSig'-b}_{\mathcal{A},S}(k)$:
  $\lambda_{ss} \xleftarrow{R} \texttt{SS.Setup}(1^k)$
  $(pk, sk) \xleftarrow{R} \texttt{SS.Kg}(\lambda_{ss})$
  $(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk)$
  $\boldsymbol{\sigma}^* \leftarrow \texttt{SS.Sign}(sk, \boldsymbol{m})$
  If $b = 0$ then
    $t' \xleftarrow{R} \mathcal{A}_2^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk, \boldsymbol{\sigma}^*)$
  Else
    $t' \xleftarrow{R} S^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk)$
  If $t' = t$ then output 1
  Else return 0

Fig. 4. Simulation-based security notions for confidentiality for (a) weakly confidential signature schemes; (b) mezzo confidential signature schemes; (c) strongly confidential signature schemes

$Expt^{\text{wHash-b}}_{\mathcal{A}}(k)$:
  $\texttt{H} \xleftarrow{R} \texttt{H.Kg}(1^k)$
  $(\boldsymbol{x}_0, t_0) \xleftarrow{R} \mathcal{A}_1(1^k)$
  $(\boldsymbol{x}_1, t_1) \xleftarrow{R} \mathcal{A}_1(1^k)$
  $\boldsymbol{h} \leftarrow \texttt{H}(\boldsymbol{x}_b)$
  $t' \xleftarrow{R} \mathcal{A}_2(1^k, \texttt{H}, \boldsymbol{h})$
  If $t' = t_0$ then output 1
  Else return 0

$Expt^{\text{sHash-b}}_{\mathcal{A}}(k)$:
  $\texttt{H} \xleftarrow{R} \texttt{H.Kg}(1^k)$
  $(\boldsymbol{x}_0, t_0) \xleftarrow{R} \mathcal{A}_1(\texttt{H})$
  $(\boldsymbol{x}_1, t_1) \xleftarrow{R} \mathcal{A}_1(\texttt{H})$
  $\boldsymbol{h} \leftarrow \texttt{H}(\boldsymbol{x}_b)$
  $t' \xleftarrow{R} \mathcal{A}_2(1^k, \texttt{H}, \boldsymbol{h})$
  If $t' = t_0$ then output 1
  Else return 0

Fig. 5. Notions of confidentiality for (a) weakly confidential hash functions; (b) strongly confidential hash functions

generation and hashing, respectively (where we usually identify the description output by the key generation algorithm $\texttt{H.Kg}$ with the hash function $\texttt{H}$ itself). The collision-finding advantage $Adv^{col}_{\mathcal{A}}$ of an attacker $\mathcal{A}$ against a hash function H is defined as

$$Adv^{col}_{\mathcal{A}}(k) = \left[\texttt{H}(x;r) = \texttt{H}(x';r') : (x, x', r, r') \xleftarrow{R} \mathcal{A}(\texttt{H});\ \texttt{H} \xleftarrow{R} \texttt{H.Kg}(1^k)\right] .$$

The hash function H is called *collision-resistant* if all PPT attacker $\mathcal{A}$ running in time t, have negligible advantage $Adv^{col}_{\mathcal{A}}(k)$ (as a function of k). We require that the hash function is hiding/confidential against an attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ playing one of the games in Figure 5. For $x \in \{w, s\}$ the attacker's advantage is defined to be

$$Adv^{\text{xHash}}_{\mathcal{A}}(k) = |\Pr[Expt^{\text{xHash-0}}_{\mathcal{A}}(k) = 1] - \Pr[Expt^{\text{xHash-1}}_{\mathcal{A}}(k) = 1]| .$$

A hash function is *weakly (resp. strongly) confidential* if every PPT attacker $\mathcal{A}$ has negligible advantage in the corresponding game subject to the following restraints:

- Pattern preserving: there exists a length function $\ell(k)$ and equality functions $\diamond_{ij} \in \{=, \neq\}$ $(1 \leq i, j \leq \ell(k))$ such that for all possible $(\boldsymbol{x}, t) \xleftarrow{R} \mathcal{A}_1(1^k)$ we have that $|\boldsymbol{x}| = \ell(k)$ and $x_i \diamond_{ij} x_j$.
- High entropy: the function $\mu(k) = \max_{x \in \{0,1\}^*} \Pr[x_i = x : (\boldsymbol{x}, t) \xleftarrow{R} \mathcal{A}_1(a)]$ is negligible for all $i \in \mathbb{N}$ and possible inputs a defined by the security notion. We define $\mu(k)$ to be the adversary's *minimum entropy.*

Note that deterministic hash functions cannot achieve strong confidentiality because an adversary $\mathcal{A}_1$ can set $t = \texttt{H}(x)$ for some message x and $\mathcal{A}_2$ can easily obtain this value from the hash vector $\boldsymbol{h}$. We also note that for "unkeyed" hash functions both notions are equivalent and so no unkeyed, deterministic hash function can be considered confidential.

In the random oracle model, where the adversary is granted oracle access to the hash function $\mathtt{H}$ instead of receiving the description as input, we give $\mathcal{A}_1$ in the strong case access to the random oracle, but deny $\mathcal{A}_1$ access to $\mathtt{H}$ in the weak case. It is easy to see that a random oracle thus achieves weak confidentiality, whereas the above attack on deterministic functions still applies in the strong case. However, under the additional constraint that $\mathcal{A}_1$ does not query $\mathtt{H}$ about any x in its output $\boldsymbol{x}$ (*hash-free adversaries*) a random oracle is also strongly confidential:

Proposition 2 (Confidentiality of Random Oracles). *For any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ where $\mathcal{A}_1$ outputs vectors of length $\ell(k)$ and with min-entropy $\mu(k)$, and where $\mathcal{A}_2$ makes at most $q_h(k)$ queries to the random oracle, we have*

$$Adv_{\mathcal{A}}^{xHash}(k) \leq 2 \cdot q_h(k) \cdot \ell(k) \cdot \mu(k)$$

for $x \in \{w, s\}$ where $\mathcal{A}$ in the strong case $x = s$ is assumed to be hash-free.

This proposition is proven in Appendix D.

As for constructions in the standard model, we note that perfectly one-way functions (POWs) [7, 8] provide a partial solution. POWs have been designed to hide all information about preimages, akin to our confidentiality notion. However, all known constructions of POWs are only good for fixed (sets of) input distributions where the distributions can depend only on the security parameter but not the hash function description. Furthermore, known POWs usually require the conditional entropy of any x_i to be high, given the other x_j's. In light of this, any $\ell(k)$-valued perfectly one-way function [8] is a weakly confidential hash function under the aforementioned restrictions. Hence, we can build such hash functions based, for example, on claw-free permutations [8] or one-way permutations [8, 14].

3.2 Full-Domain Hash Signatures

A *full-domain hash (FDH) signature scheme* FDH *for hash function* H is a signature scheme in which the signing algorithm computes a signature as $\sigma = f(\mathtt{H}(m))$ for some secret function f, and the verification algorithm checks that $g(\sigma) = \mathtt{H}(m)$ for public function g. More formally (assuming that $\mathtt{FDH.Setup}(1^k)$ outputs $\lambda_{ss} = 1^k$ and that there exists a PPT algorithm which generates the functions $(f, g) \leftarrow \mathtt{FDH.Kg}'(\lambda_{ss})$):

$\mathtt{FDH.Kg}(\lambda_{ss})$:	$\mathtt{FDH.Sign}(sk, m)$:	$\mathtt{FDH.Ver}(pk, m, \sigma)$:
$(f, g) \leftarrow \mathtt{FDH.Kg}'(\lambda_{ss})$	Parse sk as $(f, \mathtt{H})$	Parse pk as $(g, \mathtt{H})$
$\mathtt{H} \leftarrow \mathtt{H.Kg}(1^k)$	Return $\sigma = f(\mathtt{H}(m))$	Return $\top$ if $\mathtt{H}(m) = g(\sigma)$
$(pk, sk) = ((g, \mathtt{H}), (f, \mathtt{H}))$		Otherwise return $\bot$
Return (pk, sk)		

Unforgeability of FDH signatures has been shown in [5, 9]. Concerning confidentiality we prove the following proposition.

Proposition 3 (Weak Confidentiality of FDH). *The FDH-signature scheme* FDH *for hash function* H *is weakly confidential if* H *is weakly confidential. More precisely, for any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ against the weak confidentiality of* FDH*, where $\mathcal{A}_1$ outputs $\ell(k)$ messages and $\mathcal{A}_2$ makes at most $q_s(k)$ signature queries, there exists an adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ against the weak confidentiality of the hash function such that*

$$Adv_{\mathcal{A}}^{w\mathsf{FDH}}(k) \leq Adv_{\mathcal{B}}^{w\mathsf{H}}(k),$$

where $\mathcal{B}_1$'s running time is identical to the one of $\mathcal{A}_1$, and $\mathcal{B}_2$'s running time is the one of $\mathcal{A}_2$ plus $Time_{\mathtt{FDH.Kg}}(k) + (q_s + \ell(k)) \cdot Time_{\mathtt{FDH.Sign}}(k) + O(k)$.

The proof actually shows that the signature scheme remains confidential for an adversarial chosen key pair (f, g), i.e., confidentiality only relies on the confidentiality of the hash function. Moreover, by Proposition 2, we have that FDH-signature schemes are weakly confidential in the random oracle model.

Suppose $\mathsf{SS} = (\mathtt{SS.Setup}, \mathtt{SS.Kg}, \mathtt{SS.Sign}, \mathtt{SS.Ver})$ is a signature scheme. We define a new signature scheme SS' as follows (where $\mathtt{SS.Setup}' \equiv \mathtt{SS.Setup}$):

$\mathtt{SS.Kg}'(\lambda_{ss})$:	$\mathtt{SS.Sign}'(sk', m)$:	$\mathtt{SS.Ver}'(pk', m, \sigma)$:
$(pk, sk) \leftarrow \mathtt{SS.Kg}(\lambda_{ss})$	Parse sk' as $(sk, \mathtt{H})$	Parse pk' as $(pk, \mathtt{H})$
$\mathtt{H} \xleftarrow{R} \mathtt{H.Kg}(1^k)$	$r \xleftarrow{R} \{0,1\}^k$	Parse σ as (σ', r)
$pk' \leftarrow (pk, \mathtt{H})$; $sk' \leftarrow (sk, \mathtt{H})$	$h \leftarrow \mathtt{H}(r, m)$	Return $\mathtt{SS.Ver}(pk, \mathtt{H}(r, m), \sigma')$
Return (pk', sk')	$\sigma' \leftarrow \mathtt{SS.Sign}(sk, h)$	
	$\sigma \leftarrow (\sigma', r)$	
	Return σ	

Fig. 6. Construction of strongly confidential signature scheme in the ROM.

Proof. Assume that FDH is not weakly confidential and that there exists an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ successfully breaking this property. Then we construct an adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ against the weak confidentiality of the hash function as follows. Adversary $\mathcal{B}_1$ on input 1^k runs $\mathcal{A}_1$ on input 1^k and outputs this algorithm's answer $(\boldsymbol{m}, t)$.

Algorithm $\mathcal{B}_2$ receives as input a description $\mathtt{H}$ of the confidential hash function and a vector $\boldsymbol{h}$ of hash values. $\mathcal{B}_2$ runs $(f, g) \leftarrow \mathtt{FDH.Kg}'(1^k)$, sets $pk \leftarrow (g, \mathtt{H})$ and $sk \leftarrow (f, \mathtt{H})$, and computes signatures $\boldsymbol{\sigma^*} = f(\boldsymbol{h})$. It invokes $\mathcal{A}_2$ on $(1^k, pk, \boldsymbol{\sigma^*})$ and answers each subsequent signature request for message m by computing $\sigma = \mathtt{FDH.Sign}(sk, m)$. When $\mathcal{A}_2$ outputs t' algorithm $\mathcal{B}_2$ copies this output and stops.

It is easy to see that $\mathcal{B}$'s advantage attacking the confidentiality of the hash function is identical to $\mathcal{A}$'s advantage attacking the confidentiality of the FDH signature scheme (the fact that $\mathcal{A}_1$ preserves pattern and produces high-entropy messages carries over to $\mathcal{B}_1$). □

No (unforgeable) FDH-signature scheme is mezzo confidential, because a signature on the message m leaks the value $\mathtt{H}(m)$. More formally, an attacker $\mathcal{A}_1$ can pick a message $m \xleftarrow{R} \{0,1\}^k$ and set $t \leftarrow \mathtt{H}(m)$. Adversary $\mathcal{A}_2$ then receives $\sigma \leftarrow f(\mathtt{H}(m))$ and can recover $t = \mathtt{H}(m)$ by computing $g(\sigma)$. Hence, as long as finding collisions for the hash functions is hard —which is required for unforgeability— the experiment outputs 1 significantly more often in one case.

3.3 Strongly Confidential Signatures in the ROM

Recall from the previous section that FDH signatures leak the hash value of a message. To prevent this, we make the hashing process probabilistic and compute $\mathtt{H}(r, m)$ for public randomness r. Then $\mathcal{A}_1$ cannot predict the hash values of the challenge messages due to r (which becomes public only afterwards) and $\mathcal{A}_2$ cannot guess the hash values due to the entropy in the message m (even though r is then known). Our instantiation is shown in Figure 6. One can easily prove that $\mathtt{H}(r, m)$ is collision-resistant (according to our definition in Section 3.1) if we assune that (r, m) is always encoded such that one can recover r (see Appendix D.5).

Proposition 4 (Random Oracle Instantiation). *If* $\mathtt{H}$ *is a hash function modeled as a random oracle, then the signature scheme* SS' *is strongly confidential. That is, for any attacker* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *against the strong confidentiality of the signature scheme* SS'*, defined in Figure 6, where* $\mathcal{A}_1$ *outputs a vector of length* $\ell(k)$ *and with min-entropy* $\mu(k)$*, and where* $\mathcal{A}_2$ *asks at most* q_h *oracle queries (signing queries and direct hash oracle queries), we have*

$$Adv_{\mathcal{A}}^{sSig}(k) \leq 2 \cdot q_h(k) \cdot \ell(k) \cdot (2^{-k} + \mu(k)) .$$

The proof is given in Appendix D.2.

3.4 Fiat-Shamir Signature Schemes

Our second instantiation is based upon the Fiat-Shamir paradigm [13] that turns every identification scheme into a signature scheme. An identification scheme (ID scheme) is defined by a triplet

(G, S, R), where G is a key generation algorithm and the sender S wishes to prove his identity to the receiver R. More formally: $G(1^k)$ is an efficient algorithm that outputs a key pair (ipk, isk). $(S(isk), R(ipk))$ are interactive algorithms and it is required that $\Pr[(S(sk), R(pk)) = 1] = 1$ (where the probability is taken over the coin tosses of S and R). A canonical ID scheme is a 3-round ID scheme $(\alpha; \beta; \gamma)$ in which α is sent by the sender S, β by the receiver R and consists of R's random coins, and γ is sent by the sender. For a sender S with randomness r, we denote $\alpha = S(isk; r)$ and $\gamma = S(isk, \alpha, \beta; r)$. The construction is given in Figure 7. To apply the idea

Suppose (G, S, R) is a canonical identification scheme and H a hash function family. We define the signature scheme $\mathsf{SS}'' = (\mathtt{SS.Setup}'', \mathtt{SS.Kg}'', \mathtt{SS.Sign}'', \mathtt{SS.Ver}'')$ as follows (where $\mathtt{SS.Setup}(1^\lambda)$ returns $\lambda_{ss} = 1^\lambda$):

$\mathtt{SS.Kg}''(\lambda_{ss})$:
  $(ipk, isk) \leftarrow G(\lambda_{ss})$
  $\mathtt{H} \xleftarrow{R} \mathtt{H.Kg}(1^k)$
  $pk \leftarrow (ipk, \mathtt{H})$; $sk \leftarrow (isk, \mathtt{H})$
  Return (pk, sk)

$\mathtt{SS.Sign}''(sk, m)$:
  Parse sk' as $(isk, \mathtt{H})$
  $r \xleftarrow{R} \{0,1\}^k$
  $\alpha \leftarrow S(isk; r)$
  $\beta \leftarrow \mathtt{H}(\alpha, m)$
  $\gamma \leftarrow S(isk, \alpha, \beta; r)$
  $\sigma \leftarrow (\alpha, \beta, \gamma)$
  Return σ

$\mathtt{SS.Ver}''(pk', m, \sigma)$:
  Parse pk' as $(ipk, \mathtt{H})$
  Parse σ as (α, β, γ)
  $\beta' \leftarrow \mathtt{H}(\alpha, m)$
  Return 1 iff $\beta = \beta'$
    and $R(ipk, \alpha, \beta, \gamma) = 1$

Fig. 7. The Fiat-Shamir paradigm that turns every ID scheme into a signature scheme

from the plain random oracle case we need to assume that the commitment α of the Fiat-Shamir scheme has non-trivial entropy. This can always be achieved by appending public randomness.

Proposition 5 (Fiat-Shamir Instantiation). *If* $\mathtt{H}$ *is a hash function modeled as a random oracle, then the Fiat-Shamir instantiation of* SS'' *for non-trivial commitments is strongly confidential. More precisely, for any attacker* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *against the strong confidentiality of the signature scheme* SS''*, defined in Table 7, where* $\mathcal{A}_1$ *outputs a vector of length* $\ell(k)$ *and with min-entropy* $\mu(k)$*, where* α *for any pk has min-entropy* $\mu'(k)$*, and where* $\mathcal{A}_2$ *asks at most* q_h *oracle queries (signing queries and direct hash oracle queries), we have*

$$Adv_{\mathcal{A}}^{sSig}(k) \leq 2 \cdot q_h(k) \cdot \ell(k) \cdot (\mu(k) + \mu'(k)).$$

The proof is given in Appendix D.3.

3.5 Strongly Confidential Signatures from Randomness Extraction

Our instantiation in the standard model relies on randomness extractors [18, 17] and is depicted in Figure 8. The main idea is to smooth the distribution of the message via an extractor, and to sign the almost uniform value h.

Recall that a strong (a, b, n, t, ϵ)-extractor is an efficient algorithm $\mathsf{Ext} : \{0,1\}^a \times \{0,1\}^b \rightarrow \{0,1\}^n$ which takes some random input $m \in \{0,1\}^a$ (sampled according to some distribution with min-entropy at least t) and some randomness $r \in \{0,1\}^b$. It outputs $h \leftarrow \mathsf{Ext}(m, r)$ such that the statistical distance between (r, h) and pairs (r, u) for uniform random values $r \in \{0,1\}^b$ and $u \in \{0,1\}^n$ is at most ϵ.

To ensure unforgeability we need to augment the extractor's extraction property by collision-resistance, imposing the requirement that the extractors be keyed and introducing dependency of the extractor's parameters a, b, n, t, ϵ on the security parameter k (see Appendix D.5). For a survey about very efficient constructions of such collision-resistant extractors see [10].

WIn order to use extractors, we need a stronger assumption on the message distribution: we assume that the adversary $\mathcal{A}_1$ now outputs vectors of messages such that each message in the vector has min-entropy at least μ for some fixed bound $\mu(k)$ *given the other messages*. Observe that the collision-resistance requirement on the extractor implies that μ must be super-logarithmic. We say that the output has *conditional min-entropy* $\mu(k)$.

Suppose $\mathsf{SS} = (\texttt{SS.Setup}, \texttt{SS.Kg}, \texttt{SS.Sign}, \texttt{SS.Ver})$ is a signature scheme. We define a new signature scheme SS''' as follows (where $\texttt{SS.Setup}''' \equiv \texttt{SS.Setup}$):

$\texttt{SS.Kg}'''(\lambda_{ss})$:	$\texttt{SS.Sign}'''(sk', m)$:	$\texttt{SS.Ver}'''(pk', m, \sigma)$:
$(pk, sk) \leftarrow \texttt{SS.Kg}(\lambda_{ss})$	Parse sk' as $(sk, \texttt{H}, \mathsf{Ext})$	Parse pk' as $(pk, \texttt{H}, \mathsf{Ext})$
Choose an extractor Ext	$r \xleftarrow{R} \{0,1\}^b$	Parse σ as (r, σ')
$\texttt{H} \xleftarrow{R} \texttt{H.Kg}(1^k)$	$h \leftarrow \mathsf{Ext}(m, r)$	Set $h \leftarrow \mathsf{Ext}(m, r)$
$pk' \leftarrow (pk, \texttt{H}, \mathsf{Ext})$	$\sigma' \leftarrow \texttt{SS.Sign}(sk, h)$	Return $\texttt{SS.Ver}(pk, h, \sigma')$
$sk' \leftarrow (sk, \texttt{H}, \mathsf{Ext})$	$\sigma \leftarrow (\sigma', r)$	
Return (pk', sk')	Return σ	

Fig. 8. Construction of strongly confidential signature scheme based on randomness extractors.

Proposition 6 (Extractor Instantiation). *If* Ext *is an* (a, b, n, t, ϵ)*-extractor then the extractor instantiation of* SS''' *is strongly confidential. More specifically, for any attacker* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *against the strong confidentiality of the signature scheme* SS'''*, defined in Table 8, where* $\mathcal{A}_1$ *outputs a vector of length* $\ell(k)$ *with conditional min-entropy* $\mu(k) \geq t(k)$*, we have*

$$Adv_{\mathcal{A}}^{sSig}(k) \leq 2 \cdot \ell(k) \cdot \epsilon(k).$$

The proof is given in Appendix D.4.

Note that our construction of the randomness extractor operates on messages of a fixed length of $a(k)$ input bits, and the signature length depends on this value $a(k)$. To process larger messages we can first hash input messages with a collision-resistant hash function, before passing it to the extractor. In this case, some care in the analysis to determine a correct bound for the entropy lost through the hash function computation.

4 Deterministic Signcryption

Signcryption is a public-key primitive which aims to simultaneously provide message confidentiality and message integrity. Signcryption was introduced by Zheng [21] and security models were independently introduced by An, Dodis and Rabin [1] and by Baek, Steinfeld and Zheng [2]. Similar to public-key encryption, achieving confidentiality in the formal security models requires that signcryption is a randomised process; however, we may also consider the confidentiality of deterministic signcryption schemes on high-entropy message spaces. We will also see that a practical version of confidentiality may even be achieved by a deterministic signcryption scheme for low entropy message distributions. The ability to construct a deterministic signcryption scheme may be of great advantage as many signcryption schemes are based on discrete-logarithm-based digital signature schemes which are highly sensitive to imperfect randomness [16].

4.1 Notions of Confidentiality for Signcryption Schemes

A signcryption scheme is a tuple of PPT algorithms $\mathsf{SC} = (\texttt{SC.Setup}, \texttt{SC.Kg}_\texttt{s}, \texttt{SC.Kg}_\texttt{r}, \texttt{SC.SignCrypt}, \texttt{SC.UnSignCrypt})$. The setup algorithm generates public parameters $\lambda_{sc} \xleftarrow{R} \texttt{SC.Setup}(1^k)$ common to all algorithms. We will generally assume that all algorithms take λ_{sc} as an implicit input, even if it is not explicitly stated. The sender key-generation algorithm generates a key pair for the sender $(pk_S, sk_S) \xleftarrow{R} \texttt{SC.Kg}_\texttt{s}(\lambda_{sc})$ and the receiver key-generation algorithm generates a key pair for a receiver $(pk_R, sk_R) \xleftarrow{R} \texttt{SC.Kg}_\texttt{r}(\lambda_{sc})$. The signcryption algorithm takes as input a message $m \in \mathcal{M}$, the sender's private key sk_S, and the receiver's public key pk_R, and outputs a signcryption ciphertext $C \xleftarrow{R} \texttt{SC.SignCrypt}(sk_S, pk_R, m)$. The unsigncryption algorithm takes as input a ciphertext $C \in \mathcal{C}$, the sender's public key pk_S, and the receiver's private key sk_R, and outputs either a message $m \xleftarrow{R} \texttt{SC.UnSignCrypt}(pk_S, sk_R, C)$ or an error symbol $\perp$.

It is interesting to consider the basic attack on a deterministic signcryption scheme. In such an attack, the attacker picks two messages (m_0, m_1) and receives a signcryption C^* of the message m_b. The attacker checks whether C^* is the signcryption of m_0 by requesting the signcryption of m_0 from

$Expt_{\mathcal{A}}^{hSCR-b}(k)$:
$\lambda_{sc} \xleftarrow{R} \texttt{SC.Setup}(1^k)$
$(pk_S^*, sk_S^*) \xleftarrow{R} \texttt{SC.Kg}_{\texttt{s}}(\lambda_{sc})$
$(pk_R^*, sk_R^*) \xleftarrow{R} \texttt{SC.Kg}_{\texttt{r}}(\lambda_{sc})$
$(\boldsymbol{m}_0, t_0) \xleftarrow{R} \mathcal{A}_1^{\mathcal{O}}(\lambda_{sc}, pk_S^*, pk_R^*)$
$(\boldsymbol{m}_1, t_1) \xleftarrow{R} \mathcal{A}_1^{\mathcal{O}}(\lambda_{sc}, pk_S^*, pk_R^*)$
$\boldsymbol{C}^* \leftarrow \texttt{SC.SignCrypt}(\lambda_{sc}, sk_S^*, pk_R^*, \boldsymbol{m}_b)$
$t' \xleftarrow{R} \mathcal{A}_2^{\mathcal{O}}(\lambda_{sc}, pk_S^*, pk_R^*, \boldsymbol{C}^*)$
If $t' = t_0$ then output 1
Else return 0

$Expt_{\mathcal{A}}^{lSCR-b}(k)$:
$\lambda_{sc} \xleftarrow{R} \texttt{SC.Setup}(1^k)$
$(pk_S^*, sk_S^*) \xleftarrow{R} \texttt{SC.Kg}_{\texttt{s}}(\lambda_{sc})$
$(pk_R^*, sk_R^*) \xleftarrow{R} \texttt{SC.Kg}_{\texttt{r}}(\lambda_{sc})$
$(m_0, m_1, \omega) \xleftarrow{R} \mathcal{A}_1^{\mathcal{O}}(\lambda_{sc}, pk_S^*, pk_R^*)$
$C^* \leftarrow \texttt{SC.SignCrypt}(\lambda_{sc}, sk_S^*, pk_R^*, m_b)$
$b' \xleftarrow{R} \mathcal{A}_2^{\mathcal{O}}(C^*, \omega)$
Output b'

Fig. 9. Notions of confidentiality for (a) high-entropy signcryption schemes and (b) low-entropy signcryption schemes. Note that $\mathcal{A}_1$ may pass state information to $\mathcal{A}_2$ in the lSCR game. The attacker's have access to a signcryption oracle $\texttt{SC.SignCrypt}(sk_S^*, \cdot, \cdot)$ and an unsigncryption oracle $\texttt{SC.UnSignCrypt}(\cdot, sk_R^*, \cdot)$.

the signcryption oracle. As in the case of public-key encryption, we may prevent this basic attack by using a high-entropy message space and so prevent the attacker being able to determine which message to query to the signcryption oracle. However, unlike the case of public-key encryption, we may also prevent this attacker by forbidding the attacker to query the signcryption oracle on m_0 and m_1. We can therefore differentiate between the high-entropy case (in which the message distribution chosen by the attacker has high entropy) and the low-entropy case (in which the attacker is forbidden from querying the signcryption oracle on a challenge message).

We give the definition for the high-entropy and low-entropy confidentiality security notions in Figure 9. In both cases, i.e. for $x \in \{h, l\}$, the attacker's advantage is defined as

$$Adv_{\mathcal{A}}^{xSCR}(k) = |\Pr[Expt_{\mathcal{A}}^{xSCR-1} = 1] - \Pr[Expt_{\mathcal{A}}^{xSCR-0} = 1]|\,.$$

A signcryption scheme is high-entropy confidential if every PPT attacker $\mathcal{A}$ has negligible advantage in the hSCR game subject to the following restrictions:

- Strongly pattern preserving: there exists a length function $\ell(k)$, message length functions $q_i(k)$, and equality functions $\diamond_{ij} \in \{=, \neq\}$ $(1 \le i, j \le \ell(k))$ such that for all possible $(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(\lambda_{sc}, pk_S^*, pk_R^*)$ we have that $|\boldsymbol{m}| = \ell(k)$, $|m_i| = q_i(k)$ and $m_i \diamond_{ij} m_j$.
- High entropy: the function $\mu(k) = \max_{m \in \{0,1\}^*} \Pr[m_i = m : (\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(a)]$ is negligible for all $i \in \mathbb{N}$ and for any admissible input a in the corresponding game. This value is known as the adversary's minimum entropy.
- Signature free: $\mathcal{A}_1$ does not output a message $m_i \in \boldsymbol{m}$ where it has queried the signcryption oracle on the pair (pk_R^*, m_i).
- Non-trivial: $\mathcal{A}_2$ does not query the unsigncryption oracle on any pair (pk_S^*, C) where $C \in \boldsymbol{C}^*$.

A signcryption scheme is low-entropy confidential if every PPT attacker $\mathcal{A}$ has negligible advantage in the lSCR game subject to the restrictions that $\mathcal{A}$ never queries the encryption oracle on either (pk_R^*, m_0) or (pk_R^*, m_1), and $\mathcal{A}_2$ never queries the decryption oracle on (pk_S^*, C^*).

Proposition 7. *Any deterministic signcryption scheme* SC *which is low-entropy confidential is also high-entropy confidential. In particular, for any adversary* $\mathcal{A}$ *against high-entropy confidentiality, making at most* $q_s(k)$ *signcryption queries and where* $\mathcal{A}_1$ *outputs* $\ell(k)$ *messages with min-entropy* $\mu(k)$*, there exists an adversary* $\bar{\mathcal{A}}$ *such that*

$$Adv_{\mathcal{A},\mathsf{SC}}^{hSCR}(k) \le \ell(k) \cdot Adv_{\bar{\mathcal{A}},\mathsf{SC}}^{lSCR}(k) + 4 \cdot q_s(k) \cdot \ell(k) \cdot \mu(k),$$

where the running time of $\bar{\mathcal{A}}$ *equals the one of* $\mathcal{A}$ *plus* $O(k)$.

The proof essentially shows that, since the challenge messages produced by a high-entropy attacker $\mathcal{A}_1$ have min-entropy $\mu(k)$, the probability that $\mathcal{A}_2$ queries the signcryption oracle on one of those messages is bounded by $4 \cdot q_s(k) \cdot \ell(k) \cdot \mu(k)$. If this does not occur, then a low-entropy attacker can easily run a high-entropy attacker as a black-box subroutine.

We also have that the low-entropy confidentiality definition is strictly stronger than the high-entropy confidentiality definition. If SC is a high-entropy confidential signcryption scheme, then the signcryption scheme SC' given in Figure 10 is high-entropy confidential signcryption scheme but not a low-entropy confidential signcryption scheme. The proof relies on the fact that a high-entropy attacker is unlikely to output the message 0^k.

$\texttt{SC.SignCrypt}'(sk_S, pk_R, m)$:
  $C \leftarrow \texttt{SC.SignCrypt}(sk_S, pk_R, m)$
  If $m = 0^k$
    Return $C\|0$
  Else
    Return $C\|1$

$\texttt{SC.UnSignCrypt}'(pk_S, sk_R, C)$:
  Parse C as $C'\|c$ for $c \in \{0,1\}$
  $m \leftarrow \texttt{SC.UnSignCrypt}(pk_S, sk_R, C')$
  If $c = 0$ and $m \neq 0^k$
    Return $\perp$
  If $c = 1$ and $m = 0^k$
    Return $\perp$
  Else
    Return m

Fig. 10. A signcryption scheme which is high-entropy secure but not low-entropy secure

4.2 The Encrypt-and-Sign Signcryption Scheme

Initially, it may be thought that high-entropy confidentiality may be easily achieved through the combination of deterministic encryption and signature schemes. However, many of the classic composition theorems, such as encrypt-then-sign fail to achieve high-entropy security even when instantiated with secure components. Consider the encrypt-then-sign scheme, in which a signcryption is formed by first encrypting a message m with a deterministic public-key encryption scheme to give a ciphertext C, and then signing the ciphertext to obtain a signature σ. This scheme fails to achieve high-entropy confidentiality as $\mathcal{A}_1$ knows the public-key of the encryption scheme and may compute $t \leftarrow C$. $\mathcal{A}_2$ may output $t' \leftarrow C$ by inspecting the signcryption ciphertext and so "win" the security game.

However, we can show that the encrypt-and-sign (which is typically insecure as a signcryption scheme) is secure when instantiated with an IND-CCA2 public-key encryption scheme and a strongly confidential signature scheme. The construction is given in Figure 11. We will only prove the confidentiality result. The scheme can easily be shown to be unforgeable (in the sense that an attacker cannot obtain a signcryption of any message which was not previously sent by that sender to that receiver).

Proposition 8. *If the signature scheme is deterministic and strongly confidential, and the encryption scheme is* $\mathsf{IND\text{-}CCA2}$ *secure, then the signcryption scheme is confidential in the high-entropy model. In particular, if there exists a PPT attacker $\mathcal{A}$ against the high-entropy security of the signcryption scheme (asking $\ell(k)$ challenge messages), then there exists PPT attackers $\mathcal{A}_{pke}$ (resp. $\mathcal{A}_{ss}$) against the* $\mathsf{IND\text{-}CCA2}$ *security of the encryption scheme (resp. against the strong confidentiality security of the signature scheme) such that*

$$\mathsf{Adv}^{\mathrm{hSCR}}_{\mathsf{E+S},\mathcal{A}}(k) \leq \ell(k) \cdot \mathsf{Adv}^{\mathrm{cca2}}_{\mathsf{PKE},\mathcal{A}_{pke}}(k) + \mathsf{Adv}^{\mathrm{ss-conf}}_{\mathsf{SS},\mathcal{A}_{ss}}(k) \ .$$

The security of this scheme can be proven in a manner similar to the encryption/signature composition theorems proven by An *et al.* [1]. The scheme is proven secure in Appendix E.

4.3 Derandomization

Goldreich [15] presents a trick to turn any probabilistic signature scheme into a deterministic one. The idea is to include the secret key κ of a pseudorandom function $\mathsf{PRF} = (\texttt{PRF.Kg}, \texttt{PRF})$ in the

$\texttt{SC.Setup}(1^k)$
  $\lambda_{ss} \leftarrow \texttt{SS.Setup}(1^k)$
  $\lambda_{pke} \leftarrow \texttt{PKE.Setup}(1^k)$
  $\lambda_{sc} \leftarrow (\lambda_{ss}, \lambda_{pke})$
  Return (λ_{sc})

$\texttt{SC.Kg}_\texttt{r}(\lambda_{sc})$
  Parse λ_{sc} as $(\lambda_{ss}, \lambda_{pke})$
  $(pk_R, sk_R) \leftarrow \texttt{PKE.Kg}(\lambda_{pke})$
  Return (pk_R, sk_R)

$\texttt{SC.Kg}_\texttt{s}(\lambda_{sc})$
  Parse λ_{sc} as $(\lambda_{ss}, \lambda_{pke})$
  $(pk_S, sk_S) \leftarrow \texttt{SS.Kg}(\lambda_{ss})$
  Return (pk_S, sk_S)

$\texttt{SC.SignCrypt}(\lambda_{sc}, pk_R, sk_S, m)$
  Parse λ_{sc} as $(\lambda_{ss}, \lambda_{pke})$
  $c \leftarrow \texttt{PKE.Enc}(\lambda_{pke}, pk_R, (pk_S || m))$
  $\sigma \leftarrow \texttt{SS.Sign}(\lambda_{ss}, sk_S, (pk_R || m))$
  Return $C = (c, \sigma)$

$\texttt{SC.UnSignCrypt}(\lambda_{sc}, sk_R, pk_S, C)$
  Parse λ_{sc} as $(\lambda_{ss}, \lambda_{pke})$
  Parse C as (c, σ)
  $(pk'_S || m') \leftarrow \texttt{PKE.Dec}(\lambda_{pke}, sk_R, c)$
  If $pk'_S \neq pk_S$, reject
  Extract pk_R from sk_R
  If $\texttt{SS.Ver}(\lambda_{ss}, pk_S, (pk_R || m'), \sigma) = \perp$, reject
  Return m'

Fig. 11. Encrypt-and-Sign signcryption scheme.

secret signing key and, when signing a message m, using the random coins $r = \texttt{PRF}(\kappa; m)$ in this process: $\sigma = \mathcal{S}(sk, m; \texttt{PRF}(\kappa, m))$. Note that the resulting scheme now yields the same signature if run twice on the same message. A formal definition of a PRF can be found in Appendix A.

We show that Goldreich's idea applies to signcryption schemes as well, taking advantage of the fact that a signcryption scheme —as opposed to a public-key encryption scheme— involves a secret signing key in which we can put the key κ of the pseudorandom function. Nonetheless, whereas a probabilistic signcryption scheme usually hides the fact that the same message has been encrypted twice, a derandomized version clearly leaks this information.

For a signcryption scheme SC the derandomized version $\mathsf{SC}^{\mathsf{PRF}}$ based on a pseudorandom function PRF works according to Goldreich's strategy:

$\texttt{SC.Setup}^{\mathsf{PRF}}(1^k)$:
  Return $\lambda_{sc} \leftarrow \texttt{SC.Setup}(1^k)$

$\texttt{SC.Kg}_\texttt{s}{}^{\mathsf{PRF}}(\lambda_{sc})$:
  $(sk_S, pk_S) \leftarrow \texttt{SC.Kg}_\texttt{s}(\lambda_{sc})$
  $\kappa \leftarrow \texttt{PRF.Kg}(1^k)$
  $sk_S^{\mathsf{PRF}} \leftarrow (sk_S, \kappa)$; $pk_S^{\mathsf{PRF}} \leftarrow pk_S$
  Return $(sk_S^{\mathsf{PRF}}, pk_S^{\mathsf{PRF}})$

$\texttt{SC.Kg}_\texttt{r}{}^{\mathsf{PRF}}(\lambda_{sc})$:
  Return $(sk_R, pk_R) \leftarrow \texttt{SC.Kg}_\texttt{r}$

$\texttt{SC.SignCrypt}^{\mathsf{PRF}}(sk_S^{\mathsf{PRF}}, pk_R, m)$:
  parse sk_S^{PRF} as (sk_S, κ)
  $r \leftarrow \texttt{PRF}(\kappa, (pk_R, m))$
  $C \leftarrow \texttt{SC.SignCrypt}(sk_S, pk_R, m; r)$
    (i.e. using randomness r)
  Return C

$\texttt{SC.UnSignCrypt}^{\mathsf{PRF}}(sk_R, pk_S^{\mathsf{PRF}}, C)$:
  Return $\texttt{SC.UnSignCrypt}(sk_R, pk_S, C)$

Proposition 9 (Derandomized Signcryption). *Let* SC *be an unforgeable and high-entropy (resp. low-entropy) confidential signcryption scheme. Then the scheme* $\mathsf{SC}^{\mathsf{PRF}}$ *is a deterministic, unforgeable signcryption scheme which is high-entropy (resp. low-entropy) confidential. That is, for* $x \in \{l, h\}$ *and any adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *against xSCR confidentiality, there exist adversaries* $\mathcal{D}$ *and* $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ *such that*

$$\mathsf{Adv}^{\mathrm{xSCR}}_{\mathsf{SC}^{\mathsf{PRF}},\mathcal{A}}(k) \leq 2 \cdot Adv^{\mathsf{PRF}}_{\mathcal{D}}(k) + \mathsf{Adv}^{\mathrm{xSCR}}_{\mathsf{SC},\mathcal{B}}(k) + 2q_{sc}(k) \cdot \ell(k) \cdot \mu(k)$$

where $\mathcal{D}$*'s running time is identical to the one of* $\mathcal{A}$*, plus* $Time_{\texttt{SC.Setup}}(k) + Time_{\texttt{SC.Kg}_\texttt{s}}(k) + Time_{\texttt{SC.Kg}_\texttt{r}}(k) + (q_{sc} + \ell(k)) \cdot Time_{\texttt{SC.SignCrypt}}(k) + O(k)$*; the running time of* $\mathcal{B}$ *equals the one of* $\mathcal{A}$ *plus* $O(q_{sc} \cdot \log q_{sc})$.

Note that we could use the implication from low-entropy confidentiality to high-entropy confidentiality (Proposition 7) but give a direct proof to obtain better bounds. The scheme can easily be shown to be unforgeable.

Acknowledgements

The authors wish to thank the ECRYPT II MAYA working group on the design and analysis of primitives and protocols for interesting preliminary discussions on this topic. The work described in this report has in part been supported by the Commission of the European Communities through the ICT program under contract ICT-2007-216676 ECRYPT II. The information in this document is provided as is, and no warranty is given or implied that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

References

1. J. H. An, Y. Dodis, and T. Rabin. On the security of joint signature and encryption. In L. Knudsen, editor, *Advances in Cryptology – Eurocrypt 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 83–107. Springer-Verlag, 2002.
2. J. Baek, R. Steinfeld, and Y. Zheng. Formal proofs for the security of signcryption. *Journal of Cryptology*, 20(2):203–235, 2007.
3. M. Bellare, A. Boldyreva, and A. O'Neill. Deterministic and efficiently searchable encryption. In A. Menezes, editor, *Advances in Cryptology – Crypto 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 535–552. Springer-Verlag, 2007.
4. M. Bellare, M. Fischlin, A. O'Neill, and T. Ristenpart. Deterministic encryption: Definitional equivalences and constructions without random oracles. In D. Wagner, editor, *Advances in Cryptology – Crypto 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 360–378. Springer-Verlag, 2008.
5. M. Bellare and P. Rogaway. The exact security of digital signatures — how to sign with RSA and Rabin. In U. Maurer, editor, *Advances in Cryptology – Eurocrypt '96*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer-Verlag, 1996.
6. A. Boldyreva, S. Fehr, and A. O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In D. Wagner, editor, *Advances in Cryptology – Crypto 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 335–359. Springer-Verlag, 2008.
7. R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In B. Kaliski, editor, *Advances in Cryptology – Crypto '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 455–469. Springer-Verlag, 1997.
8. R. Canetti, D. Micciancio, and O. Reingold. Perfectly one-way probabilistic hash functions. In *Proc. 30th Symposium on the Theory of Computing – STOC 1998*, pages 131–140. ACM, 1998.
9. J.-S. Coron. On the exact security of full domain hash. In M. Bellare, editor, *Advances in Cryptology – Crypto 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 229–235. Springer-Verlag, 2000.
10. Y. Dodis, R. Gennaro, J. Håstad, H. Krawczyk, and T. Rabin. Randomness extraction and key derivation using the CBC, Cascade and HMAC modes. In M. Franklin, editor, *Advances in Cryptology – Crypto 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 494–510. Springer-Verlag, 2004.
11. Y. Dodis and A. Smith. Entropic security and the encryption of high entropy messages. In J. Kilian, editor, *Theory of Cryptography – TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 556–577. Springer-Verlag, 2005.
12. D. Dolev, C. Dwork, and M. Naor. Non-malleable cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.
13. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A. Odlyzko, editor, *Advances in Cryptology – Crypto '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer-Verlag, 1986.
14. M. Fischlin. Pseudorandom function tribe ensembles based on one-way permutations: Improvements and applications. In J. Stern, editor, *Advances in Cryptology - Eurocrypt 1999*, volume 1592 of *Lecture Notes in Computer Science*, pages 429–444. Springer-Verlag, 1999.
15. O. Goldreich. Two remarks concerning the Goldwasser-Micali-Rivest signature scheme. In A. M. Odlyzko, editor, *Proceedings on Advances in Cryptology – Crypto '86*, volume 263 of *Lecture Notes in Computer Science*, pages 104–110. Springer-Verlag, 1987.
16. N. A. Howgrave-Graham and N. P. Smart. Lattice attacks on digital signature schemes. *Designs, Codes and Cryptography*, 23(3):283–290, 2001.

17. N. Nisan and A. Ta-Shma. Extracting randomness: A survey and new constructions. *Journal of Computer and System Science*, 58(1):148–173, 1999.
18. N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Science*, 52(1):43–52, 1996.
19. C. Rackoff and D. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In J. Feigenbaum, editor, *Advances in Cryptology – Crypto '91*, volume 576 of *Lecture Notes in Computer Science*, pages 434–444. Springer-Verlag, 1991.
20. A. Russell and H. Wang. How to fool an unbounded adversary with a short key. In L. Knudsen, editor, *Advances in Cryptology – Eurocrypto 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 133–148. Springer-Verlag, 2002.
21. Y. Zheng. Digital signcryption or how to achieve cost(signature & encryption) ≪ cost(signature) + cost(encryption). In B. Kaliski, editor, *Advances in Cryptology – Crypto '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 165–179. Springer-Verlag, 1997.

A Standard Security Notions

A.1 Signature Schemes

The standard notion for signature security is that of (strong) existential unforgeability under chosen message attacks (sEUF-CMA). The strong version is defined below. Freshness of (m, σ) indicates that σ was never received by $\mathcal{A}$ as response to a signing request on m.

$$\mathsf{Adv}^{\mathrm{seuf-cma}}_{\mathsf{SS},\mathcal{A}}(k) = \Pr\left[\begin{array}{l}\mathtt{SS.Ver}(\lambda_{ss}, pk, m, \sigma) = \top \\ (m,\sigma) \text{ is fresh}\end{array} : \begin{array}{l}\lambda_{ss} \xleftarrow{R} \mathtt{SS.Setup}(1^k) \\ (pk, sk) \xleftarrow{R} \mathtt{SS.Kg}(\lambda_{ss}) \\ (m,\sigma) \xleftarrow{R} \mathcal{A}^{\mathtt{SS.Sign}(\lambda_{ss}, sk, \cdot)}(\lambda_{pke}, pk)\end{array}\right] .$$

The advantage $\mathsf{Adv}^{\mathrm{euf-cma}}_{\mathsf{SS},\mathcal{A}}(k)$ of the slightly weaker notion (EUF-CMA) is defined analogously, but this time m needs to be fresh (in particular, a pair (m, σ) where m was queried to the signing oracle resulting in $\sigma' \neq \sigma$ could be a valid sEUF-CMA forgery, but it is not a valid EUF-CMA one).

A.2 Public-Key Encryption

A *public key encryption* scheme is a tuple of algorithms $\mathsf{PKE} = (\mathtt{PKE.Setup}, \mathtt{PKE.Kg}, \mathtt{PKE.Enc}, \mathtt{PKE.Dec})$. First the common parameters for the given security level $k \in \mathbb{N}$ are generated by $\lambda_{pke} \xleftarrow{R} \mathtt{PKE.Setup}(1^k)$ after which a user's public/private keys are generated using $(pk, sk) \xleftarrow{R} \mathtt{PKE.Kg}(\lambda_{pke})$. Given such a key pair, a message $m \in \{0,1\}^*$ is encrypted by $c \xleftarrow{R} \mathtt{PKE.Enc}(\lambda_{pke}, pk, m)$; a ciphertext is decrypted by $m \xleftarrow{R} \mathtt{PKE.Dec}(\lambda_{pke}, sk, c)$, where possibly $\mathtt{PKE.Dec}$ outputs $\perp$ to denote an invalid ciphertext. For consistency, we require that for all $k \in \mathbb{N}$, all messages $m \in \{0,1\}^*$, it must hold that $\Pr[\mathtt{PKE.Dec}(sk, \mathtt{PKE.Enc}(pk, m)) = m] = 1$ where the probability is taken over the above randomized algorithms and $(pk, sk) \xleftarrow{R} \mathtt{PKE.Kg}(1^k)$.

The security we require for PKE is indistinguishability against chosen-ciphertext attacks IND-CCA2 security [19, 12], for which the advantage of an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is defined as

$$\mathsf{Adv}^{\mathrm{cca2}}_{\mathsf{PKE},\mathcal{A}}(k) = \left|\Pr\left[\, \mathit{Expt}^{cca2-0}_{\mathcal{A}} = 1\right] - \Pr\left[\, \mathit{Expt}^{cca-1}_{\mathcal{A}} = 1\right]\right| ,$$

where (for $b \in \{0,1\}$):

$$\begin{array}{l}
\mathit{Expt}^{cca2-b}_{\mathcal{A}} \\
\quad \lambda_{pke} \xleftarrow{R} \mathtt{PKE.Setup}(1^k) \\
\quad (pk, sk) \xleftarrow{R} \mathtt{PKE.Kg}(\lambda_{pke}) \\
\quad (m_0, m_1, \omega) \xleftarrow{R} \mathcal{A}_1^{\mathsf{PKE.Dec}(\lambda_{pke}, sk, \cdot)}(\lambda_{pke}, pk) \\
\quad c^* \xleftarrow{R} \mathtt{PKE.Enc}(\lambda_{pke}, pk, m_b) \\
\quad b' \xleftarrow{R} \mathcal{A}_2^{\mathsf{PKE.Dec}(\lambda_{pke}, sk, \cdot)}(c^*, \omega) \\
\quad \text{Output 1 if } b' = b
\end{array}$$

The adversary $\mathcal{A}_2$ is restricted not to query $\mathtt{PKE.Dec}(sk, \cdot)$ with c^*. PKE scheme PKE is said to be IND-CCA2 secure if the advantage function $\mathsf{Adv}^{\mathrm{cca2}}_{\mathsf{PKE},\mathcal{A}}(k)$ is a negligible function in k for all probabilistic polynomial-time adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

A.3 Pseudo-Random Functions

A pseudo-random function is a pair of algorithms $\mathsf{PRF} = (\mathtt{PRF.Kg}, \mathtt{PRF})$. The key generation algorithm outputs a key $\kappa \xleftarrow{R} \mathtt{PRF.Kg}(1^k)$. For our purposes, a pseudo-random function $\mathtt{PRF}(\kappa, \cdot)$ takes arbitrary bitstrings as inputs and outputs a bitstring in a given space $\mathcal{R}$. Let $\mathcal{F}$ be the set of all functions from $f : \{0,1\}^* \to \mathcal{R}$. The security of a PRF against a PPT attacker $\mathcal{A}$ is defined by the following two games:

$Expt_{\mathcal{A}}^{PRF-0}(k)$:	$Expt_{\mathcal{A}}^{PRF-1}(k)$:
$\kappa \xleftarrow{R} \mathtt{PRF.Kg}(1^k)$	$f \xleftarrow{R} \mathcal{F}$
Return $\mathcal{A}^{\mathtt{PRF}(\kappa,\cdot)}(1^k)$	Return $\mathcal{A}^{f(\cdot)}(1^k)$

The attacker's advantage is defined to be:

$$Adv_{\mathcal{A}}^{\mathtt{PRF}}(k) = |\Pr[Expt_{\mathcal{A}}^{PRF-0}(k) = 1] - \Pr[Expt_{\mathcal{A}}^{PRF-1}(k) = 1]| .$$

B Relations Between Notions of Confidentiality for Signature Schemes

In this section, we prove the separation between weakly, mezzo, and strongly confidential security. Observe that our separation results hold unconditionally in the sense that it preserves the properties of the starting scheme.

Proposition 10 (weak $\not\Rightarrow$ mezzo). *Let* SS *be a signature scheme. Then there exists a signature scheme* SS' *such that for any adversary* $\mathcal{A}'$ *against weak confidentiality of* SS' *there exists an adversary* $\mathcal{A}$ *against weak confidentiality of* SS *such that*

$$Adv_{\mathcal{A}',\mathsf{SS}'}^{wSig}(k) \le Adv_{\mathcal{A},\mathsf{SS}}^{wSig}(k) + 2 \cdot \ell(k) \cdot 2^{-k},$$

where the running time of $\mathcal{A}$ *equals the one of* $\mathcal{A}'$ *plus* $O(k)$*. Furthermore, there exists an adversary* $\mathcal{B}'$ *such that*

$$Adv_{\mathcal{B}',\mathsf{SS}'}^{mSig}(k) = 1 - 2^{-k}$$

where $\mathcal{B}'$ *runs in time* $O(k)$*.*

Proof. Take any weakly confidential confidential signature scheme $\mathsf{SS} = (\mathtt{SS.Setup}, \mathtt{SS.Kg}, \mathtt{SS.Sign}, \mathtt{SS.Ver})$ and modify it to a signature scheme SS' as follows (where $\mathtt{SS.Setup}' \equiv \mathtt{SS.Setup}$):

$\mathtt{SS.Kg}'(\lambda_{ss})$:	$\mathtt{SS.Sign}'(sk\|r, m)$:	$\mathtt{SS.Ver}'(pk\|r, m, \sigma)$:
$r \xleftarrow{R} \{0,1\}^k$	If $m = m'\|r$	If $m = m'\|r$
$(pk, sk) \xleftarrow{R} \mathtt{SS.Kg}(\lambda_{ss})$	Return $\mathtt{SS.Sign}(sk, m)\|m$	Parse σ as $\sigma'\|m$
Return $(pk\|r, sk\|r)$	Else	$\sigma \leftarrow \sigma'$
	Return $\mathtt{SS.Sign}(sk, m)$	Return $\mathtt{SS.Ver}(pk, m, \sigma)$

It follows easily that the modified scheme SS' remains weakly confidential: unless the outputs of the first stage adversary contain a message of the form $m = m'\|r$, which happens with probability at most $2 \cdot \ell(k) \cdot 2^{-k}$, any break of weak confidentiality of the derived scheme immediately yields a break of the original scheme. That is, given adversary $\mathcal{A}'$ we let $\mathcal{A}_1$ execute $\mathcal{A}_1'$ to get $(\boldsymbol{m}, t)$, and $\mathcal{A}_2$ given pk and the signatures simply appends a random string r to pk and invokes $\mathcal{A}_2'$. As long as no message in $\boldsymbol{m}$ contains r the simulation is perfect and succeeds whenever m does not contain r.

The modified scheme SS' is clearly not mezzo confidential. We can build an adversary $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ which works as follows. Algorithm $\mathcal{B}_1$ gets as input the public key $pk\|r$, chooses a message $m \xleftarrow{R} \{0,1\}^k$ at random, and sets $\boldsymbol{m} \leftarrow m\|r$ and $t \leftarrow m$. The input of the second algorithm $\mathcal{B}_2$ is a public key pk and a signature $\boldsymbol{\sigma}^*$. It parses $\boldsymbol{\sigma}^*$ as $\sigma'\|m$ and outputs m. It follows easily from the construction that the adversary $\mathcal{B}$ breaks mezzo confidentiality, in particular the advantage of $\mathcal{B}$ is

$$Adv_{\mathcal{B},\mathsf{SS}'}^{mSig}(k) = |\Pr[Expt_{\mathcal{B},\mathsf{SS}'}^{mSig-0}(k)=1] - \Pr[Expt_{\mathcal{B},\mathsf{SS}'}^{mSig-1}(k)=1]| = 1 - 2^{-k}\,.$$

$\square$

Proposition 11 (mezzo $\not\Rightarrow$ strong). *Let* SS *be a signature scheme. Then there exists a signature scheme* SS' *such that for any adversary* $\mathcal{A}'$ *against mezzo confidentiality of* SS' *there exists an adversary* $\mathcal{A}$ *against mezzo confidentiality of* SS *such that*

$$Adv_{\mathcal{A}',\mathsf{SS}'}^{mSig}(k) \le Adv_{\mathcal{A},\mathsf{SS}}^{mSig}(k) + 2\cdot\ell(k)\cdot 2^{-k},$$

where the running time of $\mathcal{A}$ *equals the one of* $\mathcal{A}'$ *plus* $O(k)$*. Furthermore, there exist a signature-free adversary* $\mathcal{B}$ *such that*

$$Adv_{\mathcal{B},\mathsf{SS}'}^{sSig}(k) = 1 - 2^{-k}$$

where $\mathcal{B}$ *runs in time* $O(k)$ *and makes a single query to its signing oracle.*

Proof. Take a mezzo confidential signature $\mathsf{SS} = (\mathtt{SS.Setup}, \mathtt{SS.Kg}, \mathtt{SS.Sign}, \mathtt{SS.Ver})$. We modify it to get a new scheme SS' as follows (where $\mathtt{SS.Setup}' \equiv \mathtt{SS.Setup}$):

$\mathtt{SS.Kg}'(\lambda_{ss})$:
  $r \xleftarrow{R} \{0,1\}^k$
  $(pk, sk) \xleftarrow{R} \mathtt{SS.Kg}(\lambda_{ss})$
  Return $(pk, sk\|r)$

$\mathtt{SS.Sign}'(sk\|r, m)$:
  Return $\mathtt{SS.Sign}(sk, m\|r)\|r$

$\mathtt{SS.Ver}'(pk, m, \sigma)$:
  Parse σ as $\sigma'\|z$ for $|z| = k$
  Return $\mathtt{SS.Ver}(pk, m\|z, \sigma)$

The scheme remains mezzo confidential because the first stage adversary can only predict r in the two executions with probability at most $2\cdot\ell(k)\cdot 2^{-k}$. This holds as, in the first stage, nothing about the value r is revealed.

The scheme SS' is clearly not strongly confidential. A successful algorithm $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ against the strong confidentiality works as follows. In the first step $\mathcal{B}_1$ takes as input the public key pk. It picks a random message $m \xleftarrow{R} \{0,1\}^k$ and invokes its signing oracle $\mathtt{SS.Sign}(sk,\cdot)$ on m in order to get a signature $\sigma = \sigma'\|r$ on m. Afterwards, $\mathcal{B}_1$ outputs $(\boldsymbol{m}, t) \leftarrow (m\|r, r)$. Note that $\mathcal{B}_1$ is signature-free because $m \neq m\|r$. The second algorithm $\mathcal{B}_2$ gets as input the tuple $(1^k, pk, \boldsymbol{\sigma})$, parses $\boldsymbol{\sigma}$ as $\sigma'\|r$ and outputs r. Obviously, $\mathcal{B}$ breaks strong confidentiality with advantage:

$$Adv_{\mathcal{B},\mathsf{SS}'}^{sSig}(k) = |\Pr[Expt_{\mathcal{B},\mathsf{SS}'}^{sSig-0}(k)=1] - \Pr[Expt_{\mathcal{B},\mathsf{SS}'}^{sSig-1}(k)=1]| = 1 - 2^{-k}\,.$$

$\square$

C Relation Between Other Notions of Confidentiality

We establish the other four relations in the next four propositions.

Proposition 12 (Balanced $xSig'$ $\Rightarrow$ **Balanced** $xSig$**).** *A scheme is* δ*-balanced xSig secure if it is* δ*-balanced* $xSig'$ *secure for some negligible value* $\delta(k)$ *where* $x \in \{w, m, s\}$*.*

Proof Let $\mathcal{A}$ be an attacker against the δ-balanced *xSig* security property of the scheme. Since $\mathcal{A}$ is also a δ-balanced $xSig'$ attacker, we have that there exists a simulator S for $\mathcal{A}$. In the δ-balanced *xSig* experiments, let T_0 be the event that $t_0 = 1$ and T_1 be the event that $t_1 = 1$. We also define $\boldsymbol{\sigma}_b^* \xleftarrow{R} \mathcal{S}(sk, \boldsymbol{m}_b)$. We can define the *xSig* advantage as shown in Figure 12. Hence, $Adv_{\mathcal{A}}^{xSig}(k)$ is negligible since $Adv_{\mathcal{A}}^{xSig'}(k)$ and δ are negligible. $\square$

Proposition 13. *A scheme is* δ*-balanced xSig secure (for some fixed* $0 \le \delta < 1/2$*) if it is 0-balanced xSig secure where* $x \in \{w, m, s\}$*.*

Proof Suppose $\mathcal{A}$ is *xSig* secure and that is δ'-balanced for some fixed $0 \le \delta' < 1/2$. We construct an attacker $\mathcal{A}'$ which is *xSig* secure and is 0-balanced. We define $\mathcal{A}' = (\mathcal{A}'_1, \mathcal{A}'_2)$ as follows:

$$
\begin{aligned}
Adv_{\mathcal{A}}^{xSig}(k) &= |\Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0] - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) = t_0]| \\
&= |\Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid T_0 \wedge T_1] \Pr[T_0 \wedge T_1] \\
&\qquad\qquad - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) = t_0 \mid T_0 \wedge T_1] \Pr[T_0 \wedge T_1] \\
&\qquad + \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid \neg T_0 \wedge T_1] \Pr[\neg T_0 \wedge T_1] \\
&\qquad\qquad - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) = t_0 \mid \neg T_0 \wedge T_1] \Pr[\neg T_0 \wedge T_1] \\
&\qquad + \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid T_0 \wedge \neg T_1] \Pr[T_0 \wedge \neg T_1] \\
&\qquad\qquad - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) = t_0 \mid T_0 \wedge \neg T_1] \Pr[T_0 \wedge \neg T_1] \\
&\qquad + \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid \neg T_0 \wedge \neg T_1] \Pr[\neg T_0 \wedge \neg T_1] \\
&\qquad\qquad - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) = t_0 \mid \neg T_0 \wedge \neg T_1] \Pr[\neg T_0 \wedge \neg T_1]| \\
&= |\Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid \neg T_0 \wedge T_1] \Pr[\neg T_0 \wedge T_1] \\
&\qquad\qquad - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) = t_0 \mid \neg T_0 \wedge T_1] \Pr[\neg T_0 \wedge T_1] \\
&\qquad + \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid T_0 \wedge \neg T_1] \Pr[T_0 \wedge \neg T_1] \\
&\qquad\qquad - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) = t_0 \mid T_0 \wedge \neg T_1] \Pr[T_0 \wedge \neg T_1]| \qquad (1) \\
&= \Pr[T_0] \Pr[\neg T_0] \cdot \\
&\qquad |\Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid \neg T_0 \wedge T_1] - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) = t_0 \mid \neg T_0 \wedge T_1] \\
&\qquad + \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid T_0 \wedge \neg T_1] - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) = t_0 \mid T_0 \wedge \neg T_1]| \qquad (2) \\
&= \Pr[T_0] \Pr[\neg T_0]| \cdot \\
&\qquad |\Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid \neg T_0 \wedge T_1] - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) \neq t_1 \mid \neg T_0 \wedge T_1] \\
&\qquad + \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid T_0 \wedge \neg T_1] - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) \neq t_1 \mid T_0 \wedge \neg T_1]| \\
&= \Pr[T_0] \Pr[\neg T_0] \cdot \\
&\qquad |\Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid \neg T_0] \; - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) \neq t_1 \mid T_1] \\
&\qquad + \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid T_0] - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) \neq t_1 \mid \neg T_1]| \qquad (3) \\
&= \Pr[\neg T_0] |\Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid T_0] \Pr[T_0] + \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid \neg T_0] \Pr[T_0] \\
&\qquad - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) \neq t_1 \mid T_1] \Pr[T_0] - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) \neq t_1 \mid \neg T_1] \Pr[T_0]| \\
&\leq \frac{1}{2} |\Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid T_0] \Pr[T_0] + \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0 \mid \neg T_0] \Pr[\neg T_0] \\
&\qquad - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) \neq t_1 \mid T_1] \Pr[T_1] - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) \neq t_1 \mid \neg T_1] \Pr[\neg T_1]| + 6\delta \qquad (4) \\
&= \frac{1}{2} |\Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_0^*) = t_0] - \Pr[\mathcal{A}_2(1^k, pk, \boldsymbol{\sigma}_1^*) \neq t_1]| + 6\delta \\
&= \frac{1}{2} |\Pr[S(1^k, pk) = t_0] - \Pr[S(1^k, pk) \neq t_1]| + 6\delta + 2Adv_{\mathcal{A}}^{xSig'}(k) \\
&= 2Adv_{\mathcal{A}}^{xSig'}(k) + 7\delta \qquad (5)
\end{aligned}
$$

Fig. 12. Bounds for $Adv_{\mathcal{A}}^{xSig'}(k)$. Equation 1 follows from the fact if $t_0 = t_1$ then the probability of $\mathcal{A}_2$ outputs t_0 is the same as the probability that it outputs t_1. Equation 2 follows from the fact that $\Pr[T_0] = \Pr[T_1]$ and that T_0 and T_1 are independent. Equation 3 follows from the fact that the probability computation no longer depends upon the value of one variable. Equation 4 follows from the fact $|\Pr[T_0] - \Pr[T_1]| \leq 2\delta$ due to the balancing property; hence, we may replace an occurrence of $\Pr[T_0]$ with $\Pr[\neg T_0]$, $\Pr[T_1]$, or $\Pr[\neg T_1]$ as long as we add a factor of 2δ. Equation 5 follows from the fact that S has no knowledge of t_0 and guesses t_0 with probability at most $1/2 + \delta$.

$\mathcal{A}_1'^{\mathcal{O}}(inp)$:
  $\beta \xleftarrow{R} \{0,1\}$
  $(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1^{\mathcal{O}}(inp)$
  If $t = \beta$ then return $(\boldsymbol{m}, t)$
  $(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1^{\mathcal{O}}(inp)$
  Return $(\boldsymbol{m}, \beta)$

$\mathcal{A}_2'^{\mathcal{S}}(1^k, pk, \boldsymbol{\sigma})$
  $t' \xleftarrow{R} \mathcal{A}_2^{\mathcal{S}}(1^k, pk, \boldsymbol{\sigma})$
  Return t'

It is clear that $\mathcal{A}'$ is 0-balanced; hence, $Adv_{\mathcal{A}'}^{xSig}(k)$ is negligible. We note that the min-entropy μ' of $\mathcal{A}'$ bounded by

$$\mu'(k) \leq \frac{2\mu(k)}{1-2\delta} + \mu$$

which is negligible since μ is negligible and δ is a fixed value.

If we examine the experiment $Expt_{\mathcal{A}'}^{xSig-b}(k)$ then $\mathcal{A}'$ is run twice. We note that the t-value produced in the second execution is ignored; hence, it is irrelevant whether the $\boldsymbol{m}_1$ is produced during the first or second execution of $\mathcal{A}$ by $\mathcal{A}'$ as the game proceeds identically in both cases. In the first execution of $\mathcal{A}'$, let E be the event that $\mathcal{A}'$ outputs the message vector $\boldsymbol{m_0}$ produced by the first execution of $\mathcal{A}$. If E does not occur, then t_0 is a (hidden) random bit and the probability $\mathcal{A}_2$ outputs t_0 is $1/2$ regardless of the bit b. If E does occur, then we are essentially playing the *xSig* security game for $\mathcal{A}$. More formally,

$$\begin{aligned}
Adv_{\mathcal{A}'}^{xSig}(k) &= |\Pr[Expt_{\mathcal{A}'}^{xSig-0}(k)=1] - \Pr[Expt_{\mathcal{A}'}^{xSig-1}(k)=1]| \\
&= |\Pr[Expt_{\mathcal{A}'}^{xSig-0}(k)=1 \mid E]Pr[E] + \Pr[Expt_{\mathcal{A}'}^{xSig-0}(k)=1 \mid \neg E]Pr[\neg E] \\
&\qquad - \Pr[Expt_{\mathcal{A}'}^{xSig-1}(k)=1 \mid E]Pr[E] - \Pr[Expt_{\mathcal{A}'}^{xSig-1}(k)=1 \mid \neg E]Pr[\neg E]| \\
&= |\Pr[Expt_{\mathcal{A}'}^{xSig-0}(k)=1 \mid E]Pr[E] - \Pr[Expt_{\mathcal{A}'}^{xSig-1}(k)=1 \mid E]Pr[E]| \\
&= Pr[E]|\Pr[Expt_{\mathcal{A}}^{xSig-0}(k)=1] - \Pr[Expt_{\mathcal{A}}^{xSig-1}(k)=1]| \\
&= Pr[E]Adv_{\mathcal{A}}^{xSig}(k) \\
&\geq (1/2-\delta)Adv_{\mathcal{A}}^{xSig}(k)\,.
\end{aligned}$$

Thus we can conclude that $Adv_{\mathcal{A}}^{xSig}(k)$ is negligible since δ is fixed and $Adv_{\mathcal{A}'}^{xSig}(k)$ is negligible. □

Proposition 14 (Balanced *xSig* $\Rightarrow$ Boolean *xSig*). *A scheme is boolean xSig secure if it is δ-balanced xSig secure for some $\delta \geq 1/p(x)$ where $x \in \{w, m, s\}$ and $p(x)$ is any polynomial.*

Proof We slightly simplify the corresponding proof by Bellare *et al.* [4]. Suppose $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be any boolean *xSig* attacker and define a $(1/p(x))$-balanced attacker $\mathcal{A}' = (\mathcal{A}_1', \mathcal{A}_2')$ as follows:

$\mathcal{A}_1'^{\mathcal{O}}(inp)$:
  $(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1^{\mathcal{O}}(inp)$
  $i \xleftarrow{R} \{1, \ldots, 2p(k)+1\}$
  If $i \in \{1, \ldots, p(k)\}$
    Return $(\boldsymbol{m}, 0)$
  If $i \in \{p(k)+1, \ldots, 2p(k)\}$
    Return $(\boldsymbol{m}, 1)$
  If $i = 2p(k)+1$
    Return $(\boldsymbol{m}, t)$

$\mathcal{A}_2'^{\mathcal{S}}(1^k, pk, \boldsymbol{\sigma})$
  $t' \xleftarrow{R} \mathcal{A}_2(1^k, pk, \boldsymbol{\sigma})$
  $j \xleftarrow{R} \{1, \ldots, 2p(k)+1\}$
  If $j \in \{1, \ldots, p(k)\}$
    Return 0
  If $j \in \{p(k)+1, \ldots, 2p(k)\}$
    Return 1
  If $j = 2p(k)+1$
    Return t'

It is easy to verify that $|\Pr[\mathcal{A}_1(inp) = 1] - 1/2| \leq 1/(2p(k)+1)$. Hence, $\mathcal{A}'$ is a $(1/p(k))$-balanced attacker. Let E be the event that $i = j = 2p(k)+1$ in the above experiment and let $\boldsymbol{\sigma}_b^* \xleftarrow{R} \mathcal{S}(sk, m_b)$.

We can compute the advantage of $\mathcal{A}'$ as:

$$
\begin{aligned}
Adv^{xSig}_{\mathcal{A}'} &= |\Pr[\mathcal{A}'_2(1^k, pk, \boldsymbol{\sigma}^*_0) = t_0] - \Pr[\mathcal{A}'_2(1^k, pk, \boldsymbol{\sigma}^*_1) = t_0]| \\
&= |\Pr[\mathcal{A}'_2(1^k, pk, \boldsymbol{\sigma}^*_0) = t_0 \mid E]\Pr[E] + \Pr[\mathcal{A}'_2(1^k, pk, \boldsymbol{\sigma}^*_0) = t_0 \mid \neg E]\Pr[\neg E] \\
&\qquad - \Pr[\mathcal{A}'_2(1^k, pk, \boldsymbol{\sigma}^*_1) = t_0 \mid E]\Pr[E] - \Pr[\mathcal{A}'_2(1^k, pk, \boldsymbol{\sigma}^*_1) = t_0 \mid \neg E] Pr[\neg E]| \\
&= \Pr[E] |\Pr[\mathcal{A}'_2(1^k, pk, \boldsymbol{\sigma}^*_0) = t_0 \mid E] - \Pr[\mathcal{A}'_2(1^k, pk, \boldsymbol{\sigma}^*_1) = t_0 \mid E]| \qquad (6) \\
&= \frac{1}{(2p(k)+1)^2} Adv^{xSig}_{\mathcal{A}}(k)
\end{aligned}
$$

Equation 6 follows from the fact that if E does not occur then either the output of $\mathcal{A}_1$ or $\mathcal{A}_2$ (or both) is an independent variable which is equal to 1 with probability 1/2 and so the $\mathcal{A}'_2$ is correct with probability 1/2 regardless of the value of $\boldsymbol{\sigma}^*$. We can conclude that $Adv^{\mathcal{A}}_{xSig}$ is negligible as $Adv^{\mathcal{A}'}_{xSig}$ is negligible (since the scheme is δ-balanced secure). $\square$

Proposition 15 (Boolean ***xSig*** $\Rightarrow$ ***xSig*****).** *A scheme is xSig-secure if it is boolean xSig-secure where* $x \in \{w, m, s\}$*.*

Proof. Consider an attacker $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ against the *xSig* secure property of the signature scheme. We define a family of boolean *xSig* attackers $\mathcal{A}^{(r)} = (\mathcal{A}^{(r)}_1, \mathcal{A}^{(r)}_2)$ with $r \in \{0,1\}^*$. Let $\langle x, y\rangle$ denote the inner product of x and y modulo 2 with the convention that x and y are padded with an appropriate number of zeroes if $|x| \neq |y|$. We define $\mathcal{A}^{(r)}$ as follows:

$\mathcal{A}^{(r)}_1(\mathit{input})$:
$(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(\mathit{input})$
$s \leftarrow \langle t, r\rangle$
Output $(\boldsymbol{m}, s)$

$\mathcal{A}^{(r)\mathcal{O}}_2(1^k, pk, \boldsymbol{\sigma})$:
$t' \xleftarrow{R} \mathcal{A}^{\mathcal{O}}_2(1^k, pk, \boldsymbol{\sigma})$
$s' \leftarrow \langle t', r\rangle$
Output s'

Since $\mathcal{A}$ is a PPT attacker we have that $|t|$ is bounded by a polynomial $p(k)$. We consider a game in which the challenger plays the boolean *xSig* game against a random attacker $\mathcal{A}^{(r)}$ where $r \xleftarrow{R} \{0,1\}^{p(k)}$. It is easy to see that $Adv^{bool}_{\mathcal{A}^{(r)}}(k) \geq \frac{1}{2} Adv^{xSig}_{\mathcal{A}}(k)$. Hence, there exists a fixed value r for which the inequality holds and this value can be hardwired into the attacker (using a non-uniform reduction). $\square$

Proposition 16 (***xSig*** $\Rightarrow$ ***xSig'*****).** *A scheme is xSig' secure if it is xSig secure where* $x \in \{w, m, s\}$*.*

Proof. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an attacker in the *xSig'* security model. We define a simulator for $\mathcal{A}$. Note that $\mathcal{A}$ is also a valid attacker in the *xSig* security model.

$S^{\texttt{SS.Sign}(sk,\cdot)}(1^k, pk)$:
$(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}^{\mathcal{O}}_1(\mathit{input})$
Parse $\boldsymbol{m}$ as $(m_1, \ldots, m_n)$
For $1 \leq i \leq n$
Query m_i to $\texttt{SS.Sign}(sk, \cdot)$ oracle and receive σ_i
Set $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_n)$
$t' \xleftarrow{R} \mathcal{A}^{\mathcal{S}(sk,\cdot)}_2(1^k, pk, \boldsymbol{\sigma})$
Output t'

An examination of the security models demonstrates that $Expt^{\text{xSig-0}}_{\mathcal{A}}(k) = Expt^{xSig'-0}_{\mathcal{A},S}(k)$ and $Expt^{\text{xSig-1}}_{\mathcal{A}}(k) = Expt^{xSig'-1}_{\mathcal{A},S}(k)$. Hence, $Adv^{xSig}_{\mathcal{A}}(k) = Adv^{xSig'}_{\mathcal{A},S}(k)$ and so the scheme is *xSig'* secure. $\square$

D Constructions of Confidential Signature Schemes

D.1 Confidentiality of Random Oracles

We begin by proving our claim about the confidentiality of random oracles. In order to do this, we first require a technical result.

Consider the advantages of two adversaries, where one runs a perfect simulation of the other one except in case of some "bad" events B_0, B_1. When the "simulation" events S_0, S_1 are related to the "experiment" events E_0, E_1 conditioned on B_0, B_1 as follows:

$$\Pr[S_0] \geq \Pr[E_0 \mid \neg B_0] \quad \text{and} \quad \Pr[\neg S_1] \geq \Pr[\neg E_1 \mid \neg B_1],$$

i.e., the simulation of experiment 0 succeeds whenever E_0 succeeds, given B_0 has not happened, and the simulation of experiment 1 fails whenever E_1 fails, given $\neg B_1$, then it holds that:

Lemma 1. *Let E_0, E_1, B_0, B_1 and S_0, S_1 be events such that*

$$\Pr[E_0] \geq \Pr[E_1] \quad \textit{and} \quad \Pr[S_0] \geq \Pr[E_0 \mid \neg B_0] \quad \textit{and} \quad \Pr[\neg S_1] \geq \Pr[\neg E_1 \mid \neg B_1].$$

Then

$$|\Pr[E_0] - \Pr[E_1]| \leq \Pr[B_0] + \Pr[B_1] + |\Pr[S_0] - \Pr[S_1]|.$$

Proof. Note that

$$\begin{aligned}
|\Pr[E_0] - \Pr[E_1]| &= \Pr[E_0] - \Pr[E_1] \\
&= \Pr[E_0] + \Pr[\neg E_1] - 1 \\
&= \Pr[E_0 \wedge B_0] + \Pr[E_0 \wedge \neg B_0] + \Pr[\neg E_1 \wedge B_1] + \Pr[\neg E_1 \wedge \neg B_1] - 1 \\
&\leq \Pr[B_0] + \Pr[B_1] + \Pr[E_0 \mid \neg B_0] + \Pr[\neg E_1 \mid \neg B_1] - 1 \\
&\leq \Pr[B_0] + \Pr[B_1] + \Pr[S_0] + \Pr[\neg S_1] - 1 \\
&\leq \Pr[B_0] + \Pr[B_1] + |\Pr[S_0] - \Pr[S_1]|.
\end{aligned}$$

□

Proposition 17 (Confidentiality of Random Oracles). *For any adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ where $\mathcal{A}_1$ outputs vectors of length $\ell(k)$ and with min-entropy $\mu(k)$, and where $\mathcal{A}_2$ makes at most $q_h(k)$ queries to the random oracle, we have*

$$Adv_{\mathcal{A}}^{xHash}(k) \leq 2 \cdot q_h(k) \cdot \ell(k) \cdot \mu(k)$$

for $x \in \{w, s\}$ where $\mathcal{A}$ in the strong case $x = s$ is assumed to be hash-free.

Proof. In the weak case the probability that $\mathcal{A}_2$ queries the random oracle in any of the at most $q_h(k)$ queries about one of the preimages of the at most $\ell(k)$ challenge values (event $\textsc{Guess}_b$), is at most $q_h(k) \cdot \ell(k) \cdot \mu(k)$ in each game. Given that $\mathcal{A}_2$ does not make such a query the distribution (over the choice of $\mathtt{H}$) of $\mathcal{A}_2$'s input —and thus of the output— in both cases $b = 0$ and $b = 1$ is independent of t_0, noting that $\mathcal{A}_1$ does not have access to the hash function. Hence, using the above lemma, the advantage is at most

$$\begin{aligned}
Adv_{\mathcal{A}}^{\mathrm{wHash}}(k) &= \left|\Pr[Expt_{\mathcal{A}}^{\mathrm{wHash}-0}(k) = 1] - \Pr[Expt_{\mathcal{A}}^{\mathrm{wHash}-1}(k) = 1]\right| \\
&\leq \Pr[\textsc{Guess}_0] + \Pr[\textsc{Guess}_1] \\
&\quad + \left|\Pr\left[Expt_{\mathcal{A}}^{\mathrm{wHash}-0}(k) = 1 \,\middle|\, \neg\textsc{Guess}_0\right] - \Pr\left[Expt_{\mathcal{A}}^{\mathrm{wHash}-1}(k) = 1 \,\middle|\, \neg\textsc{Guess}_1\right]\right| \\
&\leq 2 \cdot q_h(k) \cdot \ell(k) \cdot \mu(k).
\end{aligned}$$

In the strong case the claim follows as before, observing that $\mathcal{A}_1$ cannot make any query about the values $\boldsymbol{x_0}$ (resp. $\boldsymbol{x_1}$) by the hash freeness. It therefore holds again that (assuming $\mathcal{A}_2$ does not make a "bad" query) the input and output distribution is independent of t_0 in both cases. □

D.2 Random oracle instantiation for strongly confidential signatures

Proposition 18 (Random Oracle Instantiation). *If* $\mathtt{H}$ *is a hash function modeled as a random oracle, then the random-oracle instantiation of the signature scheme* SS' *is strongly confidential. That is, for any attacker* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *against the strong confidentiality of the signature scheme* SS'*, defined in Figure 6, where* $\mathcal{A}_1$ *outputs a vector of length* $\ell(k)$ *and with min-entropy* $\mu(k)$*, and where* $\mathcal{A}_2$ *asks at most* q_h *oracle queries (signing queries and direct hash oracle queries), we have*

$$Adv_{\mathcal{A}}^{sSig}(k) \leq 2 \cdot q_h(k) \cdot \ell(k) \cdot (2^{-k} + \mu(k)) .$$

Proof. The proof is similar to the proof of Proposition 2. There, we have observed that a random oracle is strongly confidential as long as the adversary $\mathcal{A}_1$ does not query the random oracle about one of its challenge values $\boldsymbol{m}$ (denoted as hash freeness). Here, the situation is slightly different because $\mathcal{A}_2$ does not receive signatures on the values $m_i \in \boldsymbol{m}$ directly, but signatures on randomized values $h_i \leftarrow \mathtt{H}(r_i, m_i)$. Yet, the idea of applying hash freeness carries over: Let GUESS denote the event that $\mathcal{A}_1$ queries its random oracle on one of the pairs (r_i, m_i). The probability that this event occurs is $\Pr[\,\text{GUESS}] = \ell(k) \cdot q_h(k) \cdot 2^{-k}$, where $\ell(k)$ is the length of the challenge vector and q_h denotes the number of oracle queries. In other words, we can assume that $\mathcal{A}_1$ is (quasi) hash-free.

Now consider the attacker $\mathcal{A}_2$. The probability that $\mathcal{A}_2$ queries the random oracle about any preimage of the at most $\ell(k)$ challenges is at most $\ell(k) \cdot q_h(k) \cdot \mu(k)$ in each game (because $\mathcal{A}_2$ gets r as input and the messages have entropy $\mu(k)$). Analogously to the proof of Proposition 2, we assume that $\mathcal{A}_2$ does not perform such a query. Then the distribution (over the choice of $\mathtt{H}$) of $\mathcal{A}_2$'s input, and therefore also of its output, is independent of t_0 in *both* games. Thus, we conclude that the advantage is at most

$$Adv_{\mathcal{A}}^{\mathrm{sSig}}(k) \leq 2 \cdot q_h(k) \cdot \ell(k) \cdot (2^{-k} + \mu(k)) .$$

□

D.3 Fiat-Shamir Paradigm

Proposition 19 (Fiat-Shamir Instantiation). *If* $\mathtt{H}$ *is a hash function modeled as a random oracle, then the Fiat-Shamir instantiation of* SS'' *for non-trivial commitments is strongly confidential. More precisely, for any attacker* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *against the strong confidentiality of the signature scheme* SS''*, defined in Table 7, where* $\mathcal{A}_1$ *outputs a vector of length* $\ell(k)$ *and with min-entropy* $\mu(k)$*, where* α *for any pk has min-entropy* $\mu'(k)$*, and where* $\mathcal{A}_2$ *asks at most* q_h *oracle queries (signing queries and direct hash oracle queries), we have*

$$Adv_{\mathcal{A}}^{sSig}(k) \leq 2 \cdot q_h(k) \cdot \ell(k) \cdot (\mu(k) + \mu'(k)).$$

Proof (sketch). Similar to the proof of Proposition 4 we first argue that the attacker $\mathcal{A}_1$ is quasi hash free. Recall that the commitment α has min-entropy $\mu'(k)$. Hence, the probability that $\mathcal{A}_1$ queries its random oracle about one of the challenge values $h_i \leftarrow \mathtt{H}(\alpha, m)$ (event GUESS) is $\Pr[\,\text{GUESS}] = \ell(k) \cdot q_h(k) \cdot \mu'(k)$. Assuming that $\mathcal{A}_1$ is quasi hash free, the desired bound follows analogously to the proof of Proposition 4. □

D.4 Randomness-Extractor-Based Instantiation

Proposition 20 (Extractor Instantiation). *If* Ext *is an* (a, b, n, t, ϵ)*-extractor then the extractor instantiation of* SS''' *is strongly confidential. More specifically, for any attacker* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *against the strong confidentiality of the signature scheme* SS''' *where* $\mathcal{A}_1$ *outputs a vector of length* $\ell(k)$ *with conditional min-entropy* $\mu(k) \geq t(k)$*, we have*

$$Adv_{\mathcal{A}}^{sSig}(k) \leq 2 \cdot \ell(k) \cdot \epsilon(k).$$

Proof. For the proof consider the challenge vector $\boldsymbol{m}$ that the adversary $\mathcal{A}_1$ outputs. According to our construction, each $m_i \in \boldsymbol{m}$ is executed on a randomness extractor obtaining the value $h_i \xleftarrow{R} \mathsf{Ext}(m_i; r_i)$. The attacker $\mathcal{A}_2$ then obtains a vector of signatures $\boldsymbol{\sigma}$ where the component σ_i consists of (σ'_i, r_i).

We now modify the experiment slightly substituting all the values h_i through random elements with the same bit length. Let $Expt_{\mathcal{A}}^{sSig'-b}(k)$ denote the modified experiment. Since the output of the randomness extractor is statistically close to uniform, we argue that this modification does not change the success probability of $\mathcal{A}$ too much:

$$\left|\Pr\left[\, Expt_{\mathcal{A}}^{sSig-b}(k) = 1\right] - \Pr\left[\, Expt_{\mathcal{A}}^{sSig'-b}(k) = 1\right]\right| \leq \ell(k) \cdot \epsilon(k)$$

and this holds independently of the bit b. Now, the distribution of $\mathcal{A}_2$'s input, and therefore also of its output, is independent of t_0 in *both* games. Then we can calculate the advantage of $\mathcal{A}$ as follows:

$$Adv_{\mathcal{A}}^{\mathrm{sSig}}(k) = \left|\Pr\left[\, Expt_{\mathcal{A}}^{sSig-0}(k) = 1\right] - \Pr\left[\, Expt_{\mathcal{A}}^{sSig-1}(k) = 1\right]\right| .$$

We apply the triangle inequality obtaining the desired bound:

$$\begin{aligned}
Adv_{\mathcal{A}}^{\mathrm{sSig}}(k) \leq & \left|\Pr\left[\, Expt_{\mathcal{A}}^{sSig'-0}(k) = 1\right] - \Pr\left[\, Expt_{\mathcal{A}}^{sSig'-0}(k) = 1\right]\right| \\
& + \left|\Pr\left[\, Expt_{\mathcal{A}}^{sSig'-0}(k) = 1\right] - \Pr\left[\, Expt_{\mathcal{A}}^{sSig'-1}(k) = 1\right]\right| \\
& + \left|\Pr\left[\, Expt_{\mathcal{A}}^{sSig'-1}(k) = 1\right] - \Pr\left[\, Expt_{\mathcal{A}}^{sSig'-1}(k) = 1\right]\right| \\
\leq & \; 2 \cdot \ell(k) \cdot \epsilon(k) .
\end{aligned}$$

□

D.5 Unforgeability

In this section we show that our constructions are unforgeable if the signature scheme is unforgeable and the hash function (or the extractor)is collision-resistant. Here we consider the more general case of a collision resistant function $H(r, m)$. We instantiate this function with a collision resistant hash function modeled as a random oracle (see Section 3.3), or with a collision resistant randomness extractor (see Section 3.5).

Proposition 21 (Unforgeability). *If* $\mathtt{H}$ *is a collision-resistant hash function and* SS *an unforgeable signature scheme, then the scheme* SS' *is unforgeable. More precisely, for any attacker* $\mathcal{A}'$ *against the unforgeability of* SS'*, making at most* $q_s = q_s(k)$ *signature queries, there are attackers* $\mathcal{B}$ *and* $\mathcal{A}$ *with*

$$Adv_{\mathcal{A}',\mathsf{SS}'}^{unf} \leq Adv_{\mathcal{B},\mathsf{H}}^{col}(k) + Adv_{\mathcal{A},\mathsf{SS}}^{unf}(k).$$

where $\mathcal{B}$ *has the same running time as* $\mathcal{A}'$ *plus* $O(Time_{\mathtt{SS.Setup}}(k) + Time_{\mathtt{SS.Kg}}(k) + q_s \cdot Time_{\mathtt{SS.Sign'}}(k) + q_s \cdot k)$*, and the running time of* $\mathcal{A}$ *equals the one of* $\mathcal{A}'$ *plus* $O(Time_{\mathtt{H.Kg}}(k) + q_s \cdot (Time_{\mathtt{Sample}}(k) + Time_{\mathtt{H}}(k)) + k)$.

Proof. Let $\mathcal{A}'$ be an efficient adversary against the signature scheme SS' that queries its signing oracle at most q_s times. Let (r_i, m_i) denote the corresponding pairs on which the hash function for such queries is evaluated, and m^* and r^* be the corresponding values in the forgery attempt of $\mathcal{A}'$. Also, let COLL and FORGE denote the events that $(r^*, m^*) \neq (r_i, m_i)$ for some $i \in \{1, 2, \ldots, q_s\}$, but the hash values collide, and that $\mathcal{A}'$ successfully outputs a forgery for $m^* \neq m_i$ for all $i = 1, 2, \ldots, q_s$. Then

$$Adv_{\mathcal{A}',\mathsf{SS}'}^{unf} \leq \Pr[\,\textsc{Coll}] + \Pr[\,\textsc{Forge} \mid \neg\textsc{Coll}]$$

and it remains to bound the two probabilities.

Collision-Resistance. We build an adversary $\mathcal{B}$ out of $\mathcal{A}'$, trying to find a collision for the hash function.

Setup. The input of $\mathcal{B}$ is $\mathtt{H}$. It generates a key-pair $\lambda_{ss} \leftarrow \mathtt{SS.Setup}(1^k)$; $(pk', sk') \xleftarrow{R} \mathtt{SS.Kg}(\lambda_{ss})$, sets up an initially empty query list Q, and runs black-box simulation of $\mathcal{A}'$ on input $pk = (pk', \mathtt{H})$.

Query. Whenever $\mathcal{A}'$ invokes its signing oracle on a message m, then $\mathcal{B}$ runs $(r, s) \leftarrow \mathtt{Sample}(pk)$, sets $m' \leftarrow \mathtt{H}(r, m)$, and computes the signature $\sigma' \xleftarrow{R} \mathtt{SS.Sign}(sk', m')$. It stores the tuple (r, m) in Q and returns the signature $\sigma = (\sigma', r)$.

Output. Eventually, $\mathcal{A}'$ stops, outputting a potential forgery (m^*, σ^*). $\mathcal{B}$ checks whether there exists an index $i \in \{1, 2, \ldots, q_s\}$ such that $\mathtt{H}(r^*, m^*) = \mathtt{H}(r_i, m_i)$. If so, it stops outputting $((r^*, m^*), (r_i, m_i))$, and aborts otherwise.

It follows easily from the construction that $\mathcal{B}$ achieves the claimed efficiency and that it performs a perfect simulation of the environment $\mathcal{A}'$. Hence, the advantage of $\mathcal{B}$ bounds the probability of event COLL in the attack of $\mathcal{A}'$. □

E Deterministic Signcryption Schemes

In this section, we provide the proofs of security for the deterministic signcryption schemes. We begin by showing that the low-entropy security definition implies the high-entropy security definition.

Proposition 22. *Any signcryption scheme* SC *which is low-entropy confidential is also high-entropy confidential. In particular, for any adversary* $\mathcal{A}$ *against high-entropy confidentiality, making at most* $q_s(k)$ *signcryption queries and where* $\mathcal{A}_1$ *outputs* $\ell(k)$ *messages with min-entropy* $\mu(k)$*, there exists an adversary* $\bar{\mathcal{A}}$ *such that*

$$Adv^{hSCR}_{\mathcal{A},\mathsf{SC}}(k) \leq \ell(k) \cdot Adv^{lSCR}_{\bar{\mathcal{A}},\mathsf{SC}}(k) + 4 \cdot q_s(k) \cdot \ell(k) \cdot \mu(k),$$

where the running time of $\bar{\mathcal{A}}$ *equals the one of* $\mathcal{A}$ *plus* $O(k)$.

Proof. In order to simplify the proof, we will assume that the message vector $\boldsymbol{m}$ contain distinct messages. Since the signcryption scheme is deterministic, we may always remove message duplications from the message vector $\boldsymbol{m}$ and "fill in" the corresponding ciphertexts in the ciphertext vector $\boldsymbol{C}$ by duplication.

The proof follows by a hybrid argument. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary against the high-entropy confidentiality of $\mathsf{SC} = (\mathtt{SC.Setup}, \mathtt{SC.Kg_s}, \mathtt{SC.Kg_r}, \mathtt{SC.SignCrypt}, \mathtt{SC.UnSignCrypt})$, i.e. $\mathcal{A}$ participates in the experiment $Expt^{hSCR-b}_{\mathcal{A}}(k)$ from Figure 9. We define hybrid experiments $Expt_i(k)$, $i = 1, \ldots, \ell(k)+1$ where $\ell(k) = |\boldsymbol{m}|$ for all possible $(\boldsymbol{m}, t) \xleftarrow{R} \mathcal{A}_1(\lambda_{sc}, pk^*_S, pk^*_R)$. Each experiment $Expt_i(k)$ proceeds identical to $Expt^{hSCR-0}_{\mathcal{A}}(k)$ except for the following difference in the computation of the challenge $\boldsymbol{C}^*$, i.e., for all $j = 1, \ldots, \ell(k)$:

$$\boldsymbol{C}^*[j] \leftarrow \begin{cases} \mathtt{SC.SignCrypt}(\lambda_{sc}, sk^*_S, pk^*_R, \boldsymbol{m}_1[j]) & \text{if } j < i \\ \mathtt{SC.SignCrypt}(\lambda_{sc}, sk^*_S, pk^*_R, \boldsymbol{m}_0[j]) & \text{otherwise.} \end{cases}$$

It is easy to see that $Expt_1(k) = Expt^{hSCR-0}_{\mathcal{A}}(k)$ whereas $Expt_{\ell(k)+1}(k) = Expt^{hSCR-1}_{\mathcal{A}}(k)$. Furthermore, considering the messages signcrypted in $\boldsymbol{C}^*$, these sequences trivially preserve the pattern according to $\diamond_{i,j}$.

We construct an adversary $\bar{\mathcal{A}} = (\bar{\mathcal{A}}_1, \bar{\mathcal{A}}_2)$ against the low-entropy confidentiality of SC which effectively interpolates between two subsequent hybrid experiments $Expt_i(k)$ and $Expt_{i+1}(k)$ as follows (assuming that for all $j \in [1, \ell(k)]$ messages $\boldsymbol{m}_0[j]$ and $\boldsymbol{m}_1[j]$ are distinct):

$\bar{\mathcal{A}}_1^{\mathcal{O}}(\lambda_{sc}, pk_S^*, pk_R^*)$
$(\boldsymbol{m}_0, t_0) \leftarrow \mathcal{A}_1^{\mathcal{O}}(\lambda_{sc}, pk_S^*, pk_R^*)$
$(\boldsymbol{m}_1, t_1) \leftarrow \mathcal{A}_1^{\mathcal{O}}(\lambda_{sc}, pk_S^*, pk_R^*)$
$i \xleftarrow{R} [1, \ell(k)]$
$\omega \leftarrow (\lambda_{sc}, pk_S^*, pk_R^*, i, \boldsymbol{m}_0, t_0, \boldsymbol{m}_1, t_1)$
Output $(\boldsymbol{m}_0[i], \boldsymbol{m}_1[i], \omega)$

$\bar{\mathcal{A}}_2^{\mathcal{O}}(C^*, \omega)$
Parse ω as $(\lambda_{sc}, pk_S^*, pk_R^*, i, \boldsymbol{m}_0, t_0, \boldsymbol{m}_1, t_1)$
Construct $\boldsymbol{C}^*$ as follows:
For all $j \in [1, \ell(k)]$:

$$\boldsymbol{C}^*[j] \leftarrow \begin{cases} C^* & \text{if } i = j \\ \texttt{SC.SignCrypt}(sk_S^*, pk_R^*, \boldsymbol{m}_0[j]) & \text{if } j < i \\ \texttt{SC.SignCrypt}(sk_S^*, pk_R^*, \boldsymbol{m}_1[j]) & \text{if } j > i \end{cases}$$

$t' \xleftarrow{R} \mathcal{A}_2^{\mathcal{O}}(\lambda_{sc}, pk_S^*, pk_R^*, \boldsymbol{C}^*)$
Output $t' = t_0$

Note that $\bar{\mathcal{A}}$ can easily answer signcryption and unsigncryption oracle queries of $\mathcal{A}_2$ by relaying the queries to its own oracles, as long as $\mathcal{A}_2$ does not ask $\texttt{SC.SignCrypt}(\cdot, pk_R^*, \boldsymbol{m}_0[i^*])$ or $\texttt{SC.SignCrypt}(\cdot, pk_R^*, \boldsymbol{m}_1[i^*])$ in which case $\bar{\mathcal{A}}$ aborts and outputs 0. Let Guess be the event that $\mathcal{A}_2$ makes a signcryption query (in the high-entropy game) among the at most q_s queries for any of the at most $2\ell(k)$ messages. Then we have:

$$\begin{aligned} &Adv_{\mathcal{A},\mathsf{SC}}^{\text{hSCR}}(k) \\ &\quad= \left|\Pr\left[\mathit{Expt}_{\mathcal{A},\mathsf{SC}}^{\text{hSCR-0}}(k) = 1\right] - \Pr\left[\mathit{Expt}_{\mathcal{A},\mathsf{SC}}^{hSCR-1}(k) = 1\right]\right| \\ &\quad\le 2 \cdot \Pr[\,\textsc{Guess}] \\ &\qquad\qquad + \left|\Pr\left[\mathit{Expt}_{\mathcal{A},\mathsf{SC}}^{\text{hSCR-0}}(k) = 1 \wedge \neg\textsc{Guess}\right] - \Pr\left[\mathit{Expt}_{\mathcal{A},\mathsf{SC}}^{hSCR-1}(k) = 1 \wedge \neg\textsc{Guess}\right]\right| \\ &\quad\le 4 \cdot q_s \cdot \ell(k) \cdot \mu(k) + \left|\Pr[\,\mathit{Expt}_1(k) = 1 \wedge \neg\textsc{Guess}] - \Pr\left[\mathit{Expt}_{\ell(k)+1}(k) = 1 \wedge \neg\textsc{Guess}\right]\right|. \end{aligned}$$

Let $\mathit{Expt}_j^{\bar{\mathcal{A}}-b}(k)$ denote the output of the low-entropy experiment involving $\bar{\mathcal{A}}$ and bit b, given that $\bar{\mathcal{A}}$ picks $i = j$. Taking the probability $1/\ell(k)$ for $i = j$ to happen into account, and noting that $\bar{\mathcal{A}}$ behaves identical for $b = 0$ and $b = 1$ if $\mathcal{A}$ does not trigger event Guess, we obtain:

$$\begin{aligned} &\Pr[\,\mathit{Expt}_1(k) = 1 \wedge \neg\textsc{Guess}] - \Pr\left[\mathit{Expt}_{\ell(k)+1}(k) = 1 \wedge \neg\textsc{Guess}\right] \\ &\quad= \sum_{j=1}^{\ell(k)} (\Pr\left[\mathit{Expt}_j(k) = 1 \wedge \neg\textsc{Guess}\right] - \Pr\left[\mathit{Expt}_{j+1}(k) = 1 \wedge \neg\textsc{Guess}\right]) \\ &\quad= \ell(k) \cdot \sum_{j=1}^{\ell(k)} (\Pr\left[\mathit{Expt}_j^{\bar{\mathcal{A}}-0}(k) = 1\right] - \Pr\left[\mathit{Expt}_{j+1}^{\bar{\mathcal{A}}-0}(k) = 1\right]) \\ &\quad= \ell(k) \cdot \sum_{j=1}^{\ell(k)} (\Pr\left[\mathit{Expt}_j^{\bar{\mathcal{A}}-0}(k) = 1\right] - \Pr\left[\mathit{Expt}_j^{\bar{\mathcal{A}}-1}(k) = 1\right]) \\ &\quad= \ell(k) \cdot (\Pr\left[\mathit{Expt}_{\bar{\mathcal{A}},\mathsf{SC}}^{\text{lSCR-0}}(k) = 1\right] - \Pr\left[\mathit{Expt}_{\bar{\mathcal{A}},\mathsf{SC}}^{lSCR-1}(k) = 1\right]). \end{aligned}$$

This completes the proof.

Proposition 23. *Let* SC *be a signcryption scheme. Then there exists a signcryption scheme* SC' *such that for any adversaries* $\mathcal{A}', \mathcal{B}'$ *against* SC' *there are adversaries* $\mathcal{A}, \mathcal{B}$ *against* SC *with*

$$Adv_{\mathcal{A}',\mathsf{SC}'}^{hSCR}(k) \le Adv_{\mathcal{A},\mathsf{SC}}^{hSCR}(k) + 2 \cdot \ell(k) \cdot \mu(k) \quad \text{and} \quad Adv_{\mathcal{B}',\mathsf{SC}'}^{unf}(k) \le Adv_{\mathcal{B},\mathsf{SC}}^{unf}(k)$$

where the running time of $\mathcal{A}$ *resp.* $\mathcal{B}$ *equals the one of* $\mathcal{A}'$ *resp.* $\mathcal{B}'$ *plus* $O(k)$*. Furthermore, there exists an adversary* $\mathcal{C}$ *against* SC' *with running time* $O(k)$ *such that*

$$Adv_{\mathcal{C},\mathsf{SC}'}^{lSCR}(k) = 1.$$

Proof. Take the scheme SC and modify it such that for messages $m = 0^k$ the signcryption algorithm appends 0 to the output, and 1 in any other case. That is, define SC' as follows ($\texttt{SC.Setup}' \equiv \texttt{SC.Setup}$, $\texttt{SC.Kg}_\texttt{s}{}' \equiv \texttt{SC.Kg}_\texttt{s}$ and $\texttt{SC.Kg}_\texttt{r}{}' \equiv \texttt{SC.Kg}_\texttt{r}$):

$\texttt{SC.SignCrypt}'(sk_S, pk_R, m)$:
  $C \leftarrow \texttt{SC.SignCrypt}(sk_S, pk_R, m)$
  If $m = 0^k$
    Return $C\|0$
  Else
    Return $C\|1$

$\texttt{SC.UnSignCrypt}'(pk_S, sk_R, C)$:
  Parse C as $C'\|c$ for $c \in \{0,1\}$
  $m \leftarrow \texttt{SC.UnSignCrypt}(pk_S, sk_R, C')$
  If $c = 0$ and $m \neq 0^k$
    Return $\perp$
  If $c = 1$ and $m = 0^k$
    Return $\perp$
  Else
    Return m

The fact that the derived scheme basically inherits unforgeability follows straightforwardly since one can simulate the additonal steps easily.

High-Entropy Confidentiality. We show that the derived scheme essentially preserves high-entropy confidentiality. Take an arbitrary adversary $\mathcal{A}'$ against SC', attacking the high-entropy confidentiality. Construct an adversary $\mathcal{A}$ against the underlying scheme SC as follows.

Adversary $\mathcal{A}_1$ on input $\lambda_{sc}, pk_S^*, pk_R^*$ invokes $\mathcal{A}_1'$ on these keys and runs a black-box simulation. For every query (pk_R, m) of $\mathcal{A}_1'$ to the signcryption oracle $\mathcal{A}_1$ forwards the pair to its signcryption oracle, appends 0 to the reply if $m = 0^k$ and 1 otherwise, and forwards the reply to $\mathcal{A}_1$. For every query (pk_S, C) of $\mathcal{A}_1'$ to the $\texttt{SC.UnSignCrypt}'$ oracle adversary $\mathcal{A}_1$ parses C as $C'\|c$ for $c \in \{0,1\}$. It forwards C' to $\texttt{SC.UnSignCrypt}$ to receive m and returns $\perp$ if $c = 0$ and $m \neq 0^k$, and m otherwise. Algorithm $\mathcal{A}_1$ eventually copies the output of $\mathcal{A}_1'$ and stops.

Adversary $\mathcal{A}_2$ receives as input $\lambda_{sc}, pk_S^*, pk_R^*$ and a vector $\boldsymbol{C}^*$ of signcryptions. It appends a 1-bit to each ciphertext and starts to emulate $\mathcal{A}_2'$ on the keys and the augmented ciphertexts. Algorithm $\mathcal{A}_2$ answers oracle queries as $\mathcal{A}_1$, with one exception: if $\mathcal{A}_2'$ makes a query $C\|0$ for some C in the challenge vector $\boldsymbol{C}^*$ then $\mathcal{A}_2$ returns $\perp$ without making an external oracle call.

For the analysis define the event $\textsc{Trivial}_b$ in experiment $Expt_{\mathcal{A}'}^{\text{hSCR-b}}(k)$ to occur if one of the messages in $\boldsymbol{m}_b$ equals 0^k. Note that, since the simulation of $\mathcal{A}_1'$ through $\mathcal{A}$ is perfect, the probability of event $\textsc{Trivial}_b$ happening is identical in the corresponding experiment of $\mathcal{A}$. Furthermore, given that there are no trivial messages, $\mathcal{A}$ runs a perfect simulation of $\mathcal{A}'$ in both cases $b = 0$ and $b = 1$, and the experiment of $\mathcal{A}$ succeeds (resp. fails) in this case if $\mathcal{A}'$ succeeds (resp. fails). Therefore, applying Lemma 1 from Appendix D, we obtain

$$\begin{aligned} Adv_{\mathcal{A}'}^{\text{hSCR}}(k) &\leq \left| \Pr[Expt_{\mathcal{A}'}^{hSCR-0}(k) = 1] - \Pr[Expt_{\mathcal{A}'}^{hSCR-1}(k) = 1] \right| \\ &\leq \Pr[\,\textsc{Trivial}_0] + \Pr[\,\textsc{Trivial}_1] + Adv_{\mathcal{A}}^{\text{hSCR}}(k) \\ &\leq 2 \cdot \ell(k) \cdot \mu(k) + Adv_{\mathcal{A}}^{\text{hSCR}}(k). \end{aligned}$$

Note that $\mathcal{A}_2$ never queries a challenge ciphertext C to its $\texttt{SC.UnSignCrypt}$ oracle because it sorts out queries of the form $C\|0$ by returning $\perp$ immediately, without querying its external oracle (and the challenge ciphertext $C\|1$ cannot be submitted by $\mathcal{A}'$ by assumption). Since we assume no trivial messages this behavior is identical to the one of oracle $\texttt{SC.UnSignCrypt}'$.

Low-Entropy Confidentiality. Construct the following adversary $\mathcal{C}$ against low-entropy confidentiality of SC' as follows. Adversary $\mathcal{C}_1$ outputs $m_0 = 0^k$ and $m_1 = 1^k$ and stops. Adversary $\mathcal{C}_2$ receives as input a signcryption $C = C'\|c$ for $c \in \{0,1\}$ and outputs c. It is easy to see that $\mathcal{C}_2$ predicts the bit b perfectly, yielding an advantage of 1. □

Proposition 24. *If the signature scheme is deterministic and strongly confidential, and the encryption scheme is* IND-CCA2 *secure, then the signcryption scheme is confidential in the high-entropy model. In particular, if there exists a PPT attacker $\mathcal{A}$ against the high-entropy security of* *the signcryption scheme (asking $\ell(k)$ challenge messages), then there exists PPT attackers $\mathcal{A}_{pke}$ (resp. $\mathcal{A}_{ss}$) against the* IND-CCA2 *security of the encryption scheme (resp. against the strong confidentiality security of the signature scheme) such that*

$$\mathsf{Adv}^{\mathrm{hSCR}}_{\mathsf{E+S},\mathcal{A}}(k) \leq \ell(k) \cdot \mathsf{Adv}^{\mathrm{cca2}}_{\mathsf{PKE},\mathcal{A}_{pke}}(k) + \mathsf{Adv}^{\mathrm{ss-conf}}_{\mathsf{SS},\mathcal{A}_{ss}}(k) \ .$$

Proof. Let S_i^b be the event that the adversary wins in game i^b.

Game 0^b. For $b \in \{0,1\}$ these are the experiments $Expt_{\mathcal{A}}^{hSCR-b}$ where an adversarial win is defined as the experiment outputting 1. By definition we have that

$$\mathsf{Adv}^{\mathrm{hSCR}}_{\mathsf{PKE},\mathcal{A}}(k) = \Pr\left[\, S_0^0 \,\right] - \Pr\left[\, S_0^1 \,\right] \ .$$

Game 1^b. Define Game 1^b (for $b \in \{0,1\}$) as the modification (see below) where the unsigncryption oracle checks whether the ciphertext part c of its input $C = (c, \sigma)$ corresponds to (part of) a challenge signcryption C_j. If so, there are two possibilities: either $\sigma = \sigma_j$, in which case $C \in \boldsymbol{C}^*$ violating non-triviality, or $\sigma \neq \sigma_j$. But since c_j uniquely binds m_{b_j} and pk_S^*, it follows that the unsigncryption oracle from Game 0^b would reject unless σ is a valid signature under pk_S^* for the same message $(pk_R^* || m_{b_j})$ as σ_j, this time violating the deterministic nature of the underlying signature scheme. Therefore our modification does not change the functionality of $\texttt{SC.UnSignCrypt}(\lambda_{sc}, \cdot, sk_R^*, \cdot)$ and, for $b \in \{0,1\}$, it holds that $\Pr\left[\, S_0^b \,\right] = \Pr\left[\, S_1^b \,\right]$.

$\texttt{SC.UnSignCrypt}(—, pk_S, —, C)$
  Parse C as (c, σ)
  If exists $j \in \ell(k)$ such that $(c, \sigma_j) = C_j \in \boldsymbol{C}^*$ then reject
  else
    $(pk_S' || m') \leftarrow \texttt{PKE.Dec}(\lambda_{pke}, sk_R^*, c)$
    If $pk_S' \neq pk_S$, reject
    If $\texttt{SS.Ver}(\lambda_{ss}, pk_S, (pk_R^* || m'), \sigma) = \perp$, reject
    Return m'

Game 2^b. Define Game 2^b (for $b \in \{0,1\}$) as the modification where the challenger uses encryptions of $0^{|m|}$ instead of m, but still signs m. That is the challenge oracle $\texttt{SC.SignCrypt}(\lambda_{sc}, sk_S^*, \cdot, \cdot)$ is replaced by:

$\texttt{SC.SignCrypt}'(—, —, pk_R, m)$
  $c \leftarrow \texttt{PKE.Enc}(\lambda_{pke}, pk_R, (pk_S^* || 0^{|m|}))$
  $\sigma \leftarrow \texttt{SS.Sign}(\lambda_{ss}, sk_S^*, (pk_R || m))$
  Return $C = (c, \sigma)$

We claim that for $b \in \{0,1\}$

$$\Pr\left[\, S_1^b \,\right] - \Pr\left[\, S_2^b \,\right] \leq \ell(k) \mathsf{Adv}^{\mathrm{cca2}}_{\mathsf{PKE},\mathcal{A}_{pke}'}(k)$$

and

$$\Pr\left[\, S_2^0 \,\right] - \Pr\left[\, S_2^1 \,\right] \leq \mathsf{Adv}^{\mathrm{ss-conf}}_{\mathsf{SS},\mathcal{A}_{ss}}$$

for adversaries $\mathcal{A}_{pke}'$ and $\mathcal{A}_{ss}$ described below. The claim in the proposition follows from collecting probabilities.

Justification of the hop. For concreteness we will concentrate on $b = 0$ and show that

$$\Pr\left[S_1^0\right] - \Pr\left[S_2^0\right] \leq \mathsf{Adv}^{\mathrm{cca2}}_{\mathsf{PKE},\mathcal{A}_{pke}}(k) \ .$$

for adversary $\mathcal{A}_{pke} = (\mathcal{A}_{pke_1}, \mathcal{A}_{pke_2})$ (as defined in Figure 13) against the IND-CCA2 property of the encryption scheme. The case $b = 1$ is analagous (with some obvious changes to $\mathcal{A}_{pke}$ to take into account the changed b). Note that $\mathcal{A}_{pke}$ is a multi-message IND-CCA2 adversary (asking for challenge encryption of $\ell(k)$ messages). A standard hybrid argument can be used to relate this to the IND-CCA2 advantage of a single challenge adversary $\mathcal{A}_{pke}{}'$ such that

$$\mathsf{Adv}^{\mathrm{cca2}}_{\mathsf{PKE},\mathcal{A}_{pke}}(k) \leq \ell(k)\mathsf{Adv}^{\mathrm{cca2}}_{\mathsf{PKE},\mathcal{A}_{pke}{}'}(k)$$

as used in the proposition statement.
Consider $\mathcal{A}_{pke}$ in the IND-CCA2 game $\mathit{Expt}^{cca-0}_{\mathsf{PKE},\mathcal{A}_{pke}}(k)$. In this case $\boldsymbol{c}$ will be an encryption of $\boldsymbol{m}_0$ and $\boldsymbol{C}$ will correspond to the answer to $\mathcal{A}_2$ in Game 1^0. In particular, the simulation provided by $\mathcal{A}_{pke}$ is perfect and $\mathcal{A}_2$ finds itself in Game 1^0. Note furthermore that $\mathcal{A}_{pke}$ only uses its IND-CCA2 oracle on ciphertexts not returned by its own challenge oracle.
On the other hand, if $\mathcal{A}_{pke}$ finds itself in $\mathit{Expt}^{cca-1}_{\mathsf{PKE},\mathcal{A}_{pke}}(k)$, then $\boldsymbol{c}$ will consist of a (corrupted) encryption of a matching set of zero strings and $\boldsymbol{C}$ will correspond to the answer to $\mathcal{A}_2$ in Game 2^0 and this time $\mathcal{A}_2$ finds itself in Game 2^0.
Since $\mathcal{A}_{pke}$ inherits its winning condition from $\mathcal{A}$ we have that

$$\begin{aligned}\Pr\left[S_1^0\right] - \Pr\left[S_2^0\right] &= \Pr\left[\mathit{Expt}^{cca-0}_{\mathsf{PKE},\mathcal{A}_{pke}} = 1\right] - \Pr\left[\mathit{Expt}^{cca-1}_{\mathsf{PKE},\mathcal{A}_{pke}} = 1\right] \\ &= \mathsf{Adv}^{\mathrm{cca2}}_{\mathsf{PKE},\mathcal{A}_{pke}}(k) \ .\end{aligned}$$

$\mathcal{A}_{pke_1}{}^{\texttt{PKE.Dec}(\lambda_{pke}, sk^*_R, \cdot)}(pk^*_R)$
 $\lambda_{ss} \leftarrow \texttt{SS.Setup}(1^k)$
 $(pk^*_S, sk^*_S) \leftarrow \texttt{SS.Kg}(\lambda_{ss})$
 $\lambda_{sc} \leftarrow (\lambda_{ss}, \lambda_{pke})$
 $(\boldsymbol{m}'_0, t_0) \leftarrow \mathcal{A}_1^{\mathcal{O}}(\lambda_{sc}, pk^*_S, pk^*_R)$
 $\omega \leftarrow (\lambda_{ss}, \lambda_{pke}, pk^*_R, sk^*_S, m_0, t_0)$
 For all $j \in [\ell(k)]$:
  $m_{0_j} \leftarrow (pk^*_S || m'_{0_j})$
  $m_{1_j} \leftarrow (pk^*_S || 0^{|m_{0_j}|})$
 Output $(\boldsymbol{m}_0, \boldsymbol{m}_1, \omega)$

$\mathcal{A}_{pke_2}{}^{\texttt{PKE.Dec}(\lambda_{pke}, sk^*_R, \cdot)}(\omega, \boldsymbol{c})$
 Parse ω as $(\lambda_{ss}, \lambda_{pke}, pk^*_R, sk^*_S, \boldsymbol{m}_0, t_0)$
 For all $j \in [\ell(k)]$:
  $\sigma_j \leftarrow \texttt{SS.Sign}(\lambda_{ss}, sk^*_S, (pk^*_R || m_{0_j})$
  $C_j \leftarrow (c_j, \sigma_j)$
 $t \leftarrow \mathcal{A}_2^{\mathcal{O}}(\lambda_{sc}, pk^*_S, pk^*_R, \boldsymbol{C})$
 if $t = t_0$ return 1 else return 0

$\texttt{SC.SignCrypt}(-, -, pk_R, m)$
 $c \leftarrow \texttt{PKE.Enc}(\lambda_{pke}, pk_R, (pk^*_S || m))$
 $\sigma \leftarrow \texttt{SS.Sign}(\lambda_{ss}, sk^*_S, (pk_R || m))$
 Return $C = (c, \sigma)$

$\texttt{SC.UnSignCrypt}(-, pk_S, -, C)$
 Parse C as (c, σ)
 If $c \in \boldsymbol{c}$ then reject
 Use IND-CCA2 oracle for
  $(pk'_S || m') \leftarrow \texttt{PKE.Dec}(\lambda_{pke}, sk^*_R, c)$
 If $pk'_S \neq pk_S$, reject
 Extract pk^*_R from sk^*_R
 If $\texttt{SS.Ver}(\lambda_{ss}, pk_S, (pk^*_R || m'), \sigma) = \perp$, reject
 Return m'

Fig. 13. Encrypt-and-Sign derived adversary $\mathcal{A}_{pke} = (\mathcal{A}_{pke_1}, \mathcal{A}_{pke_2})$ with oracle simulation.

Relationship with confidentiality of signature scheme. Let $\mathcal{A}_{ss} = (\mathcal{A}_{ss-1}, \mathcal{A}_{ss-2})$ as depicted in Figure 14 be the adversary against strong confidentiality of the signature scheme. It provides a perfect environment for $\mathcal{A}$ and it is clear that $\mathcal{A}_{ss-1}$ inherits the properties pattern-preserving, high entropy and signature freeness from $\mathcal{A}_1$.

$\mathcal{A}_{ss-1}^{\mathtt{SS.Sign}(\lambda_{ss}, sk_S^*, \cdot)}(pk_S^*)$
  $\lambda_{pke} \leftarrow \mathtt{PKE.Setup}(1^k)$
  $(pk_R^*, sk_R^*) \leftarrow \mathtt{PKE.Kg}(\lambda_{pke})$
  $\lambda_{sc} \leftarrow (\lambda_{ss}, \lambda_{pke})$
  $(\boldsymbol{m}, t) \leftarrow \mathcal{A}_1^{\mathcal{O}}(\lambda_{sc}, pk_S^*, pk_R^*)$
  For $j \in [\ell[k]]$:
    $m'_j \leftarrow (pk_R^* || m_j)$
  Return $(\boldsymbol{m}', t)$

$\mathcal{A}_{ss-2}^{\mathtt{SS.Sign}(\lambda_{ss}, sk_S^*, \cdot)}(pk_S^*, \boldsymbol{\sigma}^*)$
  for all $j \in \ell[k]$:
    $c_j \leftarrow \mathtt{PKE.Enc}(\lambda_{pke}, pk_R^*, (pk_S^* || 0^{q_j(k)}))$
    $C_j \leftarrow (c_j, \sigma_j)$
  $t \leftarrow \mathcal{A}_2^{\mathcal{O}}(\lambda_{sc}, pk_S^*, pk_R^*, \boldsymbol{C})$
  Return t

$\mathtt{SC.SignCrypt}(—, —, pk_R, m)$
  $c \leftarrow \mathtt{PKE.Enc}(\lambda_{pke}, pk_R, (pk_S^* || m))$
  Use $\mathtt{SS.Sign}$ oracle for
    $\sigma \leftarrow \mathtt{SS.Sign}(\lambda_{ss}, sk_S^*, (pk_R || m))$
  Return $C = (c, \sigma)$

$\mathtt{SC.UnSignCrypt}(—, pk_S, —, C)$
  Parse C as (c, σ)
  If $c \in \boldsymbol{c}$ then reject
  $(pk'_S || m') \leftarrow \mathtt{PKE.Dec}(\lambda_{pke}, sk_R^*, c)$
  If $pk'_S \neq pk_S$, reject
  If $\mathtt{SS.Ver}(\lambda_{ss}, pk_S, (pk_R^* || m'), \sigma) = \perp$, reject
  Return m'

Fig. 14. Encrypt-and-Sign derived adversary $\mathcal{A}_{ss}$.

Moreover, if $\mathcal{A}_{ss}$ finds itself in $Expt_{\mathcal{A}_{ss}}^{sSig-0}$ then $\mathcal{A}$ finds itself in Game 2^0, whereas if $\mathcal{A}_{ss}$ is in $Expt_{\mathcal{A}_{ss}}^{sSig-1}$, then $\mathcal{A}$ is playing Game 2^1. Therefore (since the winning conditions coincide):

$$\begin{aligned}\Pr\left[S_2^0\right] - \Pr\left[S_2^1\right] &= \Pr\left[Expt_{\mathcal{A}_{ss}}^{sSig-0} = 1\right] - \Pr\left[Expt_{\mathcal{A}_{ss}}^{sSig-1} = 1\right] \\ &= \mathsf{Adv}_{\mathsf{SS},\mathcal{A}_{ss}}^{\text{ss}-\text{conf}}\end{aligned}$$

This concludes the proof.

□

Proposition 25 (Derandomized Signcryption). *Let* SC *be an unforgeable and high-entropy (resp. low-entropy) confidential signcryption scheme. Then the scheme* $\mathsf{SC}^{\mathsf{PRF}}$ *is a deterministic, unforgeable signcryption scheme which is high-entropy (resp. low-entropy) confidential for distinct queries. That is, for* $x \in \{l, h\}$ *and any adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *against xSCR confidentiality, there exist adversaries* $\mathcal{D}$ *and* $\mathcal{B} = (\mathcal{B}_1, \mathcal{B}_2)$ *such that*

$$\mathsf{Adv}_{\mathsf{SC}^{\mathsf{PRF}},\mathcal{A}}^{\text{xSCR}}(k) \leq 2 \cdot Adv_{\mathcal{D}}^{\mathsf{PRF}}(k) + \mathsf{Adv}_{\mathsf{SC},\mathcal{B}}^{\text{xSCR}}(k) + 2q_{sc}(k) \cdot \ell(k) \cdot \mu(k)$$

where $\mathcal{D}$*'s running time is identical to the one of* $\mathcal{A}$*, plus* $Time_{\mathtt{SC.Setup}}(k) + Time_{\mathtt{SC.Kg_s}}(k) + Time_{\mathtt{SC.Kg_r}}(k) + (q_{sc} + \ell(k)) \cdot Time_{\mathtt{SC.SignCrypt}}(k) + O(k)$*; the running time of* $\mathcal{B}$ *equals the one of* $\mathcal{A}$ *plus* $O(q_{sc} \cdot \log q_{sc})$*. For any adversary* $\mathcal{A}$ *against unforgeability, making* q_{sc} *signcryption requests, there exist adversaries* $\mathcal{D}$ *and* $\mathcal{B}$ *such that*

$$Adv_{\mathcal{A}}^{unf}(k) \leq Adv_{\mathcal{D}}^{\mathsf{PRF}}(k) + Adv_{\mathcal{B}}^{unf}(k)$$

where $\mathcal{D}$ *runs in* $\mathcal{A}$*'s time plus* $Time_{\mathtt{SC.Setup}}(k) + Time_{\mathtt{SC.Kg_s}}(k) + Time_{\mathtt{SC.Kg_r}}(k) + (q_{sc} + \ell(k)) \cdot Time_{\mathtt{SC.SignCrypt}}(k) + O(k)$*.*

Note that we could use the implication from low-entropy confidentiality to high-entropy confidentiality (Proposition 7) but give a direct proof to obtain better bounds:

Proof. We start with the derandomized scheme $\mathsf{SC}^{\mathsf{PRF}}$ and show that any unforgeability or confidentiality attacker can be turned into one against the original probabilistic scheme SC with essentially the same success probability. We start with the case of high-entropy confidentiality; the other cases follow below.

High-Entropy Confidentiality. Assume, in a thought experiment, that we replace the pseudorandom function $\mathtt{PRF}$ in the scheme $\mathsf{SC}^{\mathsf{PRF}}$ by a truly random function. Denote this scheme by $\mathsf{SC}^{\textsc{Rnd}}$. Note that this scheme would not be efficiently implementable but it only serves as an intermediate step. We claim that the advantage of any adversary attacking $\mathsf{SC}^{\mathsf{PRF}}$ in the high-entropy confidential game is at most the advantage of attacking $\mathsf{SC}^{\textsc{Rnd}}$ plus the advantage of distinguishing the pseudorandom function $\mathtt{PRF}$ from a truly random function $\textsc{Rnd}$:

$$\mathsf{Adv}^{\mathrm{hSCR}}_{\mathsf{SC}^{\mathsf{PRF}},\mathcal{A}}(k) \leq \mathsf{Adv}^{\mathrm{hSCR}}_{\mathsf{SC}^{\textsc{Rnd}},\mathcal{A}}(k) + 2 \cdot Adv^{\mathsf{PRF}}_{\mathcal{D}}(k)$$

This can be seen as follows. Construct a distinguisher $\mathcal{D}$ with oracle access to either an instance $\mathtt{PRF}(\kappa, \cdot)$ for a random key κ, or to a truly random function $\textsc{Rnd}(\cdot)$, via a black-box simulation of $\mathcal{A}$. To be more precise, we actually consider two distinguishers $\mathcal{D}_b$ with a bit b hardwired, determining which game $\mathcal{D}$ simulates; but since the two distinguishers behave identical we comprise them in one algorithm. Algorithm $\mathcal{D}$ (with bit $b \in \{0, 1\}$) simulates all steps in the high-entropy game for b, except that for signature queries by $\mathcal{A}$ and for creating the challenge signcryptions, $\mathcal{D}$ calls its oracle to create the randomness from the message. Distinguisher $\mathcal{D}$ eventually runs the check of the experiment and outputs the corresponding bit.

Clearly, the advantage of $\mathcal{A}$ attacking $\mathsf{SC}^{\textsc{Rnd}}$ instead of $\mathsf{SC}^{\mathsf{PRF}}$ cannot drop by more than twice the distinguishing advantage of $\mathcal{D}$, where the factor two originates from the case of two distinguishers. We next argue that, in the experiment involving the scheme $\mathsf{SC}^{\textsc{Rnd}}$, with high probability the signcryption algorithm is never run on the same pair (pk_R, m) twice (including the step where the challenge ciphertexts are created). Here we assume that we first "normalize" $\mathcal{A}$ in the sense that none of the challenge messages are identical. This can be easily fixed by removing such entries from $\mathcal{A}_1$'s output and duplicating signcryption entries in $\mathcal{A}_2$'s input with the help of the $\diamond_{ij}$ relation. Denote the event that the signcryption algorithm in an attack for bit b is run on the same input again by $\textsc{twice}_b$.

- Since we demand signature-freeness, $\mathcal{A}_1$ never outputs a message for which it has called the signcryption oracle before.
- The query distinctiveness guarantees that $\mathcal{A}$ never calls the signcryption oracle twice about the same message-key pair.

Hence, the only case that a signcryption query for (pk_R, m) can be made twice, is that $\mathcal{A}_2$ calls the oracle about a message m output by $\mathcal{A}_1$. By the high-entropy assumption, though, the probability that this happens for any of the $\ell(k)$ messages output by $\mathcal{A}_1$ in any of the at most $q_{\mathrm{sc}}(k)$ signcryption queries, is at most

$$\Pr[\,\textsc{twice}_b] \leq \ell(k) \cdot q_{\mathrm{sc}}(k) \cdot \mu(k)$$

independently of the bit b.

Given that event $\textsc{twice}_b$ does not occur, the random function $\textsc{Rnd}$ generates a fresh random string for each signcryption run —as the probabilistic scheme, too, would. But then it follows that the advantage of attacking $\mathsf{SC}^{\textsc{Rnd}}$ compared to the one of attacking the original probabilistic scheme differs by at most $\Pr[\,\textsc{twice}_0] + \Pr[\,\textsc{twice}_1]$ (cf. Lemma 1). The claim now follows.

Low-Entropy Confidentiality. The low-entropy case is almsot identical to the high-entropy case. Only here the adversary $\mathcal{A}_2$ is explicitly forbidden to ask the signcryption oracle from either of the two challenge messages m_0, m_1, thuis ensuring that algorithm is never executed on the same pair (pk_R, m) twice. The claim then follows analogously.

Unforgeability. Unforgeability of the derandomized version follows as in the original transformation by Goldreich [15], noting that we only rely on the pseudorandomness of $\mathtt{PRF}$. That is, with the same step as in the proof for confidentiality one can show that $\mathcal{A}$'s success probability attacking $\mathsf{SC}^{\mathsf{PRF}}$ and $\mathsf{SC}^{\textsc{Rnd}}$ can only differ in the distinguishing advantage against the pseudorandom function. Then, in another step, one can build an adversary $\mathcal{B}$ against the underlying (probabilistic) scheme

SC which relays the communication between $\mathcal{A}$ and the signcryption oracle, but re-injects previous replies for identical queries. The success probability of $\mathcal{B}$ is identical to the one $\mathcal{A}$ attacking SC^{Rnd}, plus the time $O(kq_{\text{sc}} \cdot \log q_{\text{sc}})$ to maintain the list of previous queries. $\square$