# Signatures with Post-Compromise Accountability

Dennis Dayanikli[1], Johannes Lang[2⋆], and Anja Lehmann[1]

[1] Hasso Plattner Institute, University of Potsdam
{dennis.dayanikli, anja.lehmann}@hpi.de
[2] Technische Universität Dresden
johannes.lang@tu-dresden.de

**Abstract.** Cryptographic signatures play an integral part in ensuring authenticity and integrity in digital systems. Their security crucially relies on the secrecy of the signing key, since knowledge of this key enables an adversary to generate valid signatures on any message. Once a signing key is compromised, the standard countermeasure is to revoke the corresponding public key and to invalidate all signatures produced for this key. However, with this approach even legitimate signatures created by the honest signer would retroactively lose their validity. In this work, we initiate the formal study of a new approach – *Signatures with Post-Compromise Accountability (SPCA)* – which provides security guarantees even after the secret key was compromised. This notion effectively introduces a grace period for the legitimate key owner, during which the validity of honestly generated signatures is preserved despite the adversary's knowledge of the secret key. We formally define SPCA and its security guarantees, and present two constructions achieving this notion. Our first construction generalizes the signature-in-signature approach of Błaśkiewicz et al. (ESORICS '21), where an inner signature is embedded into the randomness of an outer signature. This construction, however, requires revealing the signing secret key during revalidation. Our second construction overcomes this limitation by enabling revalidation without disclosing the secret key, yielding stronger security guarantees.

## 1 Introduction

Digital signatures are a fundamental building block to ensure authenticity and integrity. They secure software updates, authenticate financial transactions, and are used in countless communication protocols. The core security requirement for digital signatures is existential unforgeability under adaptive chosen-message attacks. This notion ensures that an attacker with access only to the public key and signatures on messages of its choice cannot generate a forgery, i.e., a new message-signature pair that verifies correctly. In classical schemes, this guarantee holds only as long as the signing key remains secret – once the key is exposed, there is no unforgeability guarantee anymore.

⋆ Research conducted while being a Master's student at HPI.

*Key Compromise & Mitigation Strategies.* Compromising signing keys is not a rare edge case but a recurring reality. In practice, signing keys may leak through software vulnerabilities, malware, side-channel attacks, or human factors such as coercion and mishandling of credentials [3,23,39,41]. Large-scale incidents – ranging from stolen code-signing certificates [29] to compromised government PKIs [18] – demonstrate that even carefully protected keys can fall into adversarial hands. Moreover, the advent of quantum computers threatens many widely deployed signature schemes [45], turning every long-term secret key into a future liability. This motivates the design of signature schemes that retain meaningful security guarantees even after compromise.

The typical mitigation against long-term effects of key compromise is to allow the signer to update or rotate its key so that, after detection of compromise, future signatures regain their security guarantees. This paradigm of key-evolving signatures [37] has been studied in the form of forward-secure [1,2,4,24,30], monotone [40], key-insulated [14,15,35,36] or intrusion-resilient [26,34] signatures. Another mitigation strategy is to distribute the signing capability across multiple parties using threshold or multi-signatures [9,13,25,28]. In this setting, an adversary must compromise all, or at least a certain threshold, of keys in order to forge signatures, which significantly reduces the impact of individual key compromise. While such mechanisms reduce the risk of forgery through key compromise, they do not eliminate it entirely. In particular, advances in cryptanalysis – most notably quantum algorithms – may allow powerful adversaries to recover long-term secret keys regardless of protective measures.

Further, a crucial gap remains: signatures issued between the compromise and its detection are all considered untrustworthy, as there are no mechanisms to decide whether those signatures originate from the honest signer or from an adversary. In fact, this creates a time where all authentication guarantees are lost: signatures are deemed invalid retroactively, if the compromise is detected with a delay (which is very common, e.g. [38]), and obviously no new key material has been set then as well.

*Post-Compromise Accountability Through Extra Hidden Keys.* The recent work of Błaśkiewicz et al. [7] sketches an interesting solution to preserve accountability guarantees for signatures in a post-compromise scenario. Their construction enables a verifier to determine the legitimate origin of signatures, even when the adversary has compromised the signing key. This is achieved by introducing an additional *accountable key pair* whose existence remains entirely hidden from both verifiers and adversaries during normal operation. Only once the signer becomes aware of a key compromise does it reveal the accountable verification key to the verifier. Under this model, an adversary mounting a conventional key-compromise attack is assumed to obtain only the primary signing key, while the accountable secret key remains secret. Since no information about the accountable keys is ever revealed through the standard signing process, this construction also rules out attacks – classical or quantum – that rely on extracting information about the accountable keys from observed signatures.

Błaśkiewicz et al. [7] outline this idea through a "signature-in-signature" construction. Therein, an *inner signature*, generated using the accountable secret key, is embedded into the randomness of an *outer signature* created with the main signing key. These outer signatures remain computationally indistinguishable from standard ones, ensuring that the hidden layer cannot be targeted, as no information about it is leaked. Moreover, the main signature scheme maintains compatibility with existing verification algorithms. Once the signer becomes aware that its secret signing key has been compromised, it discloses the accountable verification key. A verifier can then extract the inner signature and verify it against the accountable verification key, thereby distinguishing authentic signatures from forgeries even after the signing key has been compromised. In this way, the scheme restores accountability for the period during which the adversary was in possession of the signing key.

Their approach, which they instantiate with Schnorr signatures, shows a promising idea for achieving post-compromise accountability but lacks a formal security model and analysis. Moreover, in order to revalidate signatures, their scheme requires the signer to not only disclose its accountable verification key to the verifier but also its main signing key. The verifier needs this signing key to extract the hidden inner signature. Although leaking the signing key may be acceptable under the assumption that is has already been compromised, this requirement is clearly suboptimal. It forces key disclosure even in scenarios where compromise is uncertain, e.g., if an adversary deceives the signer into revealing it, and may be infeasible in practice. In particular, if the signing key is protected by secure hardware, the honest signer may be unable to extract its own key, even though a more powerful adversary could compromise the key via side-channel or quantum attacks.

*Motivation for Accountable Keys.* At first glance, assuming that accountable keys cannot be compromised may seem overly strong, raising the question of why they are not simply used as ordinary signing keys. The crucial distinction lies in how accountable keys are used: their existence and use are deliberately hidden from adversaries. In contrast, classic digital signatures rely on a public verification key that is known and reused over long periods of time, giving an adversary ample opportunity to mount targeted attacks aimed at recovering the corresponding secret key. Since the accountable verification key remains hidden, an adversary cannot even identify a target, let alone attack it.

### 1.1 Our Contribution

Our work formalizes and improves the post-compromise accountability idea put forward by Błaśkiewicz et al. [7], addressing the lack of a formal security model and the practical limitation of disclosing the signing key for revalidation. We present generic constructions for achieving post-compromise accountability that generalize the approach of Błaśkiewicz et al. In addition, we give concrete instantiations of our framework and demonstrate that the Schnorr-in-Schnorr construction proposed by Błaśkiewicz et al. is insecure.

*Security Model* We introduce a formal security model for Signatures with Post-Compromise Accountability (SPCA) which closely follows the idea of [7]. In our model, a signer has both a signing key pair and an accountable key pair. The signer generates *accountable signatures* using both its signing key and an accountable secret key. The accountable secret key is intended to reside in a hardened component (e.g., secure hardware or a trusted execution environment) and is therefore assumed to be significantly harder to compromise than the main signing key. Accordingly, we consider adversaries that may compromise the main signing key but are unable to compromise the accountable secret key. Accountable signatures can be verified with a standard public key, but also support revalidation after key compromise. To enable revalidation, the signer reveals its accountable verification key and its secret signing key. As argued above, the exposure of the signing key might not be feasible for the honest user, and is a dangerous design choice in general. Therefore, we introduce an enhanced variant SPCA$^+$, where the signer does not need to publish its signing key. We define the following security properties for both variants which we formalize in a game-based security model.

**Pre-Compromise Unforgeability** captures security prior to compromise and matches the guarantees of standard signature schemes: an adversary without the signing key cannot forge signatures that pass standard verification, and access to accountable keys must not weaken this guarantee.

**Post-Compromise Accountability** captures security after compromise: an adversary who has compromised the signing key but not the accountable secret key should be unable to forge accountable signatures that pass revalidation. This property acts as a fallback mechanism to preserve the security of signatures that were honestly generated between key compromise and its detection. Accordingly, we allow a lower security level for post-compromise accountability. To compensate for this relaxation, it is crucial that no information about accountable keys is revealed prior to compromise; this requirement motivates the indistinguishability property presented next.

**Indistinguishability** ensures that post-compromise accountability is not weakened by information leaked before compromise. In particular, no information about the accountable keys must be revealed prior to compromise. To formalize this requirement, we introduce a reference signing algorithm that operates solely on the main signing key. We then require that accountable signatures are indistinguishable from signatures produced by this reference algorithm. In practice, this reference algorithm typically corresponds to a standard signing procedure and may even conceal the use of SPCA entirely.

**Accountable-Key Committing** ensures the authenticity of the accountable verification key that is published by the signer after compromise detection. The verifier establishes trust in the supplied key by revalidating previously generated accountable signatures with the provided key. For this mechanism to be sound, accountable signatures must commit to a unique accountable verification key, a property we formalize as accountable-key committing.

*Constructions & Instantiations* We present two generic constructions: $\mathsf{SiS}$, which securely realizes SPCA, and $\mathsf{SiS}^+$, which realizes SPCA$^+$. Both constructions generalize the Schnorr-specific approach of [7] where an inner signature is used as randomness for an outer signature scheme. We specify requirements on the inner and outer scheme and provide instantiations for both, in the random oracle model. Furthermore, we show that instantiating the outer scheme with Schnorr signatures fails to achieve pre-compromise unforgeability. Consequently, the construction of Błaśkiewicz et al. [7] does not satisfy our formal security model.

### 1.2 Related Work

As discussed in our motivation, prior work has addressed the risks of key compromise through mechanisms such as distributed trust and key-evolving signatures. Here, we highlight works that are most closely related to our technical approach and to the notion of post-compromise accountability.

*Incoercible signatures* [16,22,31] address threats from active coercion by allowing a signer to produce signatures that appear genuine but can later be identified as fake by a trusted authority. While similar in spirit to our use of signatures with indistinguishable extra structure, our approach does not rely on a trusted authority and instead leverages honestly generated signatures prior to compromise.

*Anamorphic signatures* [12,27,33] enable encrypted communication in settings where only signatures are allowed, typically by using the randomness of probabilistic signature schemes as a covert channel. This technique is also used in our $\mathsf{SiS}$ and $\mathsf{SiS}^+$ constructions, following the framework of Jaeger and Stracovsky [27]. Unlike anamorphic signatures, which embed arbitrary messages, our goal is to embed an additional unforgeable signature on the same message. Relatedly, Kutyłowski et al. [32] study covert signatures embedded in public ledgers. While similar in technique, their work does not provide a formal security model and does not address post-compromise accountability.

*Electronic seals* [6] cryptographically chain signatures to contextual data, enhancing auditability and regulatory compliance, but rely on stateful updates and trusted authorities, and cannot distinguish genuine from forged signatures once a key leaks. Other directions, such as blockchain-based signatures [46], reduce the impact of compromise by supporting rapid key revocation through public ledgers. While these methods mitigate or contain the effects of compromise, they do not ensure post-compromise accountability.

## 2 Preliminaries

With $\lambda, \kappa$ we denote security parameters. We denote by $y \leftarrow \mathcal{A}(x)$ the execution of algorithm $\mathcal{A}$ on input $x$ with output $y$. By $s \xleftarrow{r} S$, we mean that $s$ is chosen uniformly at random from the set $S$.

A hash function is a deterministic mapping $H : \{0,1\}^* \to \{0,1\}^\lambda$. It is said to be collision-resistant if no efficient adversary can find two distinct inputs $x \neq x'$

such that $H(x) = H(x')$ with more than negligible probability. In some settings, $H$ is modeled as a random oracle, meaning that each new input is mapped to a uniformly random value in the output space, and the only way to evaluate $H$ is by querying an oracle.

A signature scheme consists of three PPT algorithms $\mathsf{KGen}(1^\lambda) \to (sk, pk)$, $\mathsf{Sign}(sk, m) \to \sigma$ and $\mathsf{Vf}(pk, m, \sigma) \to 0/1$. The standard property for signatures is existential unforgeability under chosen messages (UNF-CMA), which requires that no PPT adversary can produce a valid signature on a fresh message with non-negligible probability. We briefly recall the Schnorr signature scheme which will be used in some of our constructions, and whose security is based on the discrete logarithm assumption.

**Definition 1 (Schnorr Signature).** *Let $(\mathbb{G}, g, q)$ define a cyclic group $\mathbb{G}$ of prime order $q$ with generator $g$ and let $H : \{0,1\}^* \to \mathbb{Z}_p$ be a hash function. The Schnorr signature scheme $\Pi := (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vf})$ with message space $\mathcal{M} = \{0,1\}^*$ and signature space $\Sigma = \mathbb{Z}_q \times \mathbb{Z}_q$ is defined as follows:*

$\mathsf{KGen}(1^\lambda)$: $sk \xleftarrow{r} \mathbb{Z}_q$. $pk := g^{sk}$. Return $(sk, pk)$.
$\mathsf{Sign}(sk, m)$: $r \xleftarrow{r} \mathbb{Z}_q$, $e := H(m, g^r)$, $s := r + sk \cdot e \bmod q$. Return $\sigma := (e, s)$.
$\mathsf{Vf}(pk, m, \sigma)$: Parse $\sigma := (e, s)$. Output 1 if $e = H(m, g^s \cdot pk^{-e})$.

## 3 Signatures with Post-Compromise Accountability

In this section, we introduce *Signatures with Post-Compromise Accountability* (SPCA), a strengthened notion of digital signatures that incorporates an additional mechanism to preserve accountability even in the event of key compromise. We first define the syntax and corresponding algorithms, and then formalize the relevant security properties.

### 3.1 Syntax

SPCA is parameterized by two security parameters, $\lambda$ and $\kappa$. Intuitively, $\lambda$ represents the default (pre-compromise) security level, while $\kappa$ describes the fallback post-compromise security level. The parameter $\lambda$ corresponds to the conventional security level used e.g. in signature schemes, and should therefore be chosen accordingly. By contrast, if the time between compromise and detection is assumed to be short, it is acceptable to target a lower post-compromise security level $\kappa$.

In $\mathsf{KGen}$, both a signing key pair $(sk, pk)$ and an accountable key pair $(ask, avk)$ are generated. Signing is performed via $\mathsf{AccSign}$, which uses both $sk$ and $ask$ to produce an *accountable signature*. There are two distinct verification algorithms: Until a compromise is detected, the verifier uses algorithm $\mathsf{Vf}$ to verify signatures with $pk$. If the user suspects its key $sk$ was compromised, it reveals $avk$ to the verifier who can then revalidate all signatures – particularly those created between the moment of compromise and its detection – using $\mathsf{Reval}$. This way, the verifier can determine whether a signature is an accountable signature that

was generated with $(sk, ask)$ or forged only with knowledge of $sk$. We distinguish two variants of SPCA. In basic SPCA, revalidation algorithm $\mathsf{Reval}$ requires $pk$, $avk$ and $sk$, which means the user must also reveal its $sk$ to the verifier upon suspected compromise. In SPCA$^+$, the algorithm $\mathsf{Reval}$ only requires $pk$ and $avk$, thus avoiding disclosure of $sk$.

**Definition 2 (Signatures with Post-Compromise Accountability).** *A signature scheme with post-compromise accountability* $\boxed{\mathsf{SPCA}}/\mathsf{SPCA}^+ = (\mathsf{KGen}, \mathsf{AccSign}, \mathsf{Vf}, \mathsf{Reval})$ *with message space* $\mathcal{M}$ *and signature space* $\Sigma$ *is defined as follows:*

$\mathsf{KGen}(1^\lambda, 1^\kappa) \to (sk, pk, ask, avk)$: *Takes security parameters* $1^\lambda$ *and* $1^\kappa$ *and outputs a signing key pair* $(sk, pk)$ *and an accountable key pair* $(ask, avk)$.

$\mathsf{AccSign}(sk, ask, m) \to \sigma$: *Generates an accountable signature* $\sigma$ *on message* $m \in \mathcal{M}$ *using* $sk$ *and* $ask$.

$\mathsf{Vf}(pk, m, \sigma) \to 0/1$: *Outputs 1 if* $\sigma$ *is a signature on* $m$ *which verifies under* $pk$, *and* 0 *otherwise.*

$\mathsf{Reval}(\boxed{sk,}pk, avk, m, \sigma) \to 0/1$: *Outputs 1 if* $\sigma$ *is a valid accountable signature on* $m$ *under* $pk$ *and* $avk$, *and* 0 *otherwise.* $\mathsf{SPCA}$ *additionally takes* $\boxed{sk}$ *as input.*

*Correctness.* An SPCA scheme must be correct, meaning that all honestly generated accountable signatures verify under both regular verification and revalidation. Specifically, for all $(sk, pk, ask, avk) \leftarrow \mathsf{KGen}(1^\lambda, 1^\kappa)$ it must hold that $\mathsf{Vf}(pk, m, \mathsf{AccSign}(sk, ask, m)) = 1 \wedge \mathsf{Reval}(sk, pk, avk, m, \mathsf{AccSign}(sk, ask, m)) = 1$. Similarly, for SPCA$^+$ it must hold that $\mathsf{Vf}(pk, m, \mathsf{AccSign}(sk, ask, m)) = 1 \wedge \mathsf{Reval}(pk, avk, m, \mathsf{AccSign}(sk, ask, m)) = 1$.

### 3.2 Unforgeability & Accountability

The primary security goal for SPCA and SPCA$^+$ is that only a party holding the appropriate secret key(s) can generate valid signatures. This is captured via unforgeability; however, we distinguish between the two verification algorithms used before and after compromise. **Pre-compromise unforgeability** refers to unforgeability with respect to the verification algorithm $\mathsf{Vf}$ used before compromise, and **post-compromise accountability** refers to unforgeability with respect to revalidation algorithm $\mathsf{Reval}$ used after compromise.

We describe both of these properties for SPCA; the definition for SPCA$^+$ is analogous, with the only difference that $\mathsf{Reval}$ does not require $sk$. As a shorthand, we write SPCA$^{(+)}$ to refer to SPCA and SPCA$^+$ simultaneously.

**Pre-Compromise Unforgeability** Prior to any compromise, security of SPCA should match that of a conventional signature scheme: an adversary who does not know the signing key $sk$ should not be able to produce signatures that verify under the standard verification algorithm $\mathsf{Vf}$. Importantly, the introduction of

accountable keys must not weaken this guarantee; in particular, access to the accountable keys must not help an adversary forge signatures.

We formalize pre-compromise unforgeability via a security experiment closely following the classical UNF-CMA experiment for signatures, but adapted to account for the additional key pair $(ask, avk)$. In our experiment, the adversary has access to an $\mathsf{AccSign}$ oracle that produces honestly generated signatures using both $sk$ and $ask$. As one of our goals is seamless integration of SPCA into existing signature deployments, pre-compromise unforgeability should be at least as strong as classical unforgeability with respect to $\mathsf{Vf}$ and $pk$. In particular, the introduction of $(ask, avk)$ must not weaken the classical guarantees. To capture this, the adversary is generously given the accountable key pair $(ask, avk)$ in the experiment and we require this property to hold at security level $\lambda$. With this modeling, the security thus relies solely on the secrecy of $sk$. The $\mathsf{AccSign}$ oracle still uses $ask$ since accountable signatures are jointly generated with $sk$ and $ask$. If the adversary cannot produce a valid forgery on a fresh message under these conditions with more than negligible probability, the scheme is pre-compromise unforgeable. The formal experiment is depicted in Fig. 1.

**Definition 3 (Pre-Compromise Unforgeability).** *A $SPCA^{(+)}$-scheme $\mathsf{SPCA}^{(+)}$ is pre-compromise unforgeable if for any PPT adversary $\mathcal{A}$ there exists a negligible function $\mathsf{negl}$ such that $\Pr[\mathsf{Exp}^{\mathsf{preUNF}}_{\mathcal{A},\mathsf{SPCA}}(\lambda,\kappa)=1] \leq \mathsf{negl}(\lambda)$ holds.*

**Post-Compromise Accountability** After compromise, the revalidation algorithm $\mathsf{Reval}$ is used to distinguish signatures created by the honest signer – using both the signing key and the accountable secret key – from signatures produced using only the compromised signing key. Accordingly, an adversary who knows the signing key but not the accountable secret key must be unable to generate signatures that revalidate under $\mathsf{Reval}$.

We formalize post-compromise accountability also via an unforgeability experiment (Fig. 1). Again, the challenger creates the user's keys $(sk, pk, ask, avk)$ and grants the adversary access to an $\mathsf{AccSign}$ oracle. Post-compromise accountability should hold as long as $ask$ remains secret. Thus, the adversary is generously given all other keys in the experiment, namely $(sk, pk, avk)$. The adversary's objective is to produce a fresh accountable signature that verifies under $\mathsf{Reval}$ with respect to $(sk, pk, avk)$. The scheme achieves post-compromise accountability if no efficient adversary can succeed with more than negligible probability, and this guarantee is required to hold at the fallback security level $\kappa$. Because this notion permits a reduced security level, it is essential that no information about accountable keys is revealed prior to compromise. This requirement is captured by the indistinguishability property defined next.

**Definition 4 (Post-Compromise Accountability).** *A $SPCA^{(+)}$-scheme $\mathsf{SPCA}^{(+)}$ is post-compromise accountable if for any PPT adversary $\mathcal{A}$ there exists a negligible function $\mathsf{negl}$ such that $\Pr[\mathsf{Exp}^{\mathsf{postACC}}_{\mathcal{A},\mathsf{SPCA}}(\lambda,\kappa)=1] \leq \mathsf{negl}(\kappa)$ holds.*

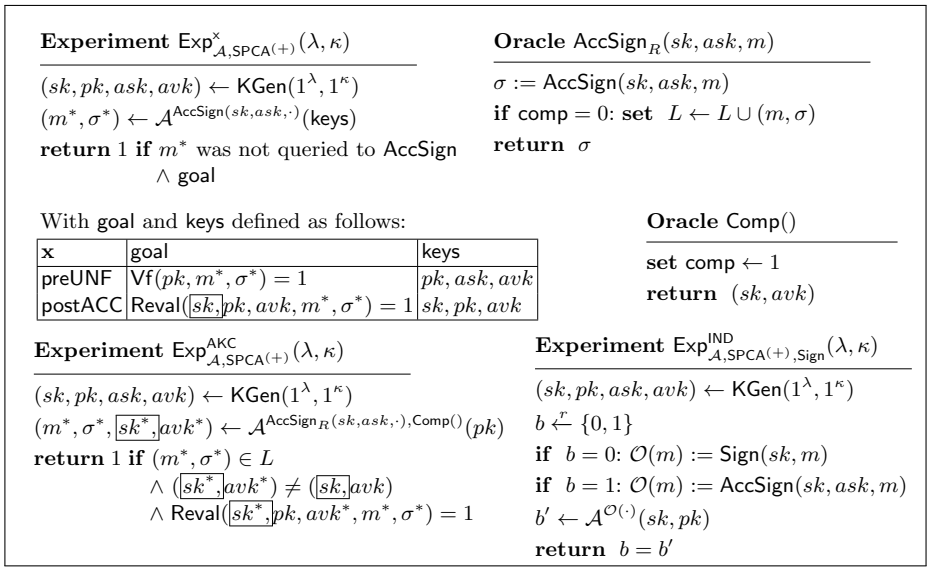



Fig. 1: Security experiments for SPCA (including boxed text) and $\mathsf{SPCA}^+$.

### 3.3 Indistinguishability

We allow a reduced security level for post-compromise accountability because it only needs to protect the limited time window between compromise and detection. This relaxation is acceptable since we additionally require that, prior to compromising a signer, the adversary obtains no information about the accountable keys. In particular, the adversary should not be able to identify or target the accountable key in advance. We formalize this requirement as *indistinguishability* of accountable signatures.

In practice, an adversary may observe accountable signatures generated by the signer using $ask$. To ensure that these signatures do not reveal any information about $avk$, we require them to be indistinguishable from signatures that can be produced using the signing key $sk$ alone. For this purpose, we introduce a reference signing algorithm Sign that operates solely on $sk$ and outputs signatures that verify under Vf for $pk$. Ideally, Sign corresponds to a standard signing algorithm, which may even conceal the existence of SPCA entirely.

We formalize indistinguishability via a security experiment. In this experiment, the adversary has access to an oracle that either returns accountable signatures generated by $\mathsf{AccSign}(sk,ask,m)$ or signatures generated by $\mathsf{Sign}(sk,m)$. The adversary additionally receives $(sk,pk)$, modeling that even after compromising $sk$, it should not learn any information about $avk$ from accountable signatures. The scheme is indistinguishable if no efficient adversary can distinguish between these two cases.

**Definition 5 (Indistinguishability).** *A $SPCA^{(+)}$ scheme $\mathsf{SPCA}^{(+)}$ is indistinguishable with respect to signing algorithm $\mathsf{Sign}(sk,m)$ if for any PPT adver-*

*sary $\mathcal{A}$ there exists a negligible function* $\mathsf{negl}$ *such that* $\Pr[\mathsf{Exp}^{\mathsf{IND}}_{\mathcal{A},\mathsf{SPCA},\mathsf{Sign}}(\lambda,\kappa) = 1] \leq \frac{1}{2} + \mathsf{negl}(\kappa)$ *holds.*

### 3.4 Accountable-Key Committing

Upon detecting a key compromise, the signer publishes its accountable verification key (and the signing key in SPCA) to enable revalidation via $\mathsf{Reval}$. This allows a verifier to distinguish signatures produced by the legitimate signer using both $sk$ and $ask$ from signatures created after compromise using $sk$ alone (potentially using reference algorithm $\mathsf{Sign}$). For accountability to be meaningful, the verifier must obtain the signer's *authentic* $avk$ after compromise detection. To establish trust in the provided $avk$, the verifier can rely on honestly generated accountable signatures created before compromise as a trust anchor: we require that every honest accountable signature commits to a unique pair $(sk, avk)$, so that no alternative key pair $(sk', avk')$ can revalidate the same signature. A verifier can then choose a signature generated prior to compromise – ideally the oldest available one – and accept the submitted keys only if this signature successfully revalidates under $(sk, avk)$.

We formalize this requirement as *accountable-key committing* via a security experiment (Fig. 1). The adversary is given the public key $pk$ and oracle access to honestly generated accountable signatures. It also has access to a $\mathsf{Comp}$ oracle that reveals the user's keys $(sk, avk)$. The adversary wins if it outputs a distinct pair $(sk^*, avk^*) \neq (sk, avk)$ under which an honestly generated accountable signature successfully revalidates. If no efficient adversary can do so, the scheme satisfies accountable-key committing. We require this property to hold at the fallback security level $\kappa$, reflecting that the accountable key space is of $\kappa$ size, so that guessing $avk$ succeeds only with probability negligible in $\kappa$.

*Accountable-Key Committing for SPCA$^+$.* In SPCA$^+$, revalidation depends only on $avk$ rather than $(avk, sk)$. Hence, accountable-key committing for SPCA$^+$ requires that the adversary cannot produce an alternative $avk^* \neq avk$ under which an honest signature revalidates. The security experiment in Fig. 1 also defines the SPCA$^+$ setting.

**Definition 6 (Accountable-Key Committing).** *A SPCA$^{(+)}$-scheme* $\mathsf{SPCA}^{(+)}$ *is accountable-key committing if for any PPT adversary $\mathcal{A}$ there exists a negligible function* $\mathsf{negl}$ *such that* $\Pr[\mathsf{Exp}^{\mathsf{AKC}}_{\mathcal{A},\mathsf{SPCA}}(\lambda,\kappa) = 1] \leq \mathsf{negl}(\kappa)$ *holds.*

*Accountable-Key Committing vs. Indistinguishability.* At first sight, accountable-key committing and indistinguishability appear to be conflicting goals. Indistinguishability demands that signatures leak no information about the accountable keys, whereas accountable-key committing requires that every signature uniquely commits to a particular $avk$. The two properties are, however, compatible once their scope is distinguished: indistinguishability must hold for all parties that do not know $avk$, in which case accountable signatures are computationally indistinguishable from standard ones. Once $avk$ is disclosed, accountable signatures

become distinguishable, and accountable-key committing ensures that they can be unambiguously linked to the correct accountable key.

*No Non-Repudiation.* While indistinguishability between accountable and regular signatures is required to safeguard the accountable keys, it also introduces a limitation: SPCA cannot guarantee meaningful non-repudiation, i.e., it cannot prevent a signer from later denying having generated a valid signature on a given message. A malicious user could always choose to sign a message using algorithm $\mathsf{Sign}(sk, m)$ instead of $\mathsf{AccSign}(sk, ask, m)$, and later deny responsibility by claiming they did not sign the message. Consequently, the only guarantee SPCA can provide is the following: if the verifier determines that a signature is a valid accountable signature, then it must have originated from the legitimate user. By contrast, if a signature passes standard verification under $pk$ but not the revalidation, it is ambiguous – such a signature could have been generated either by an adversary who compromised $sk$ or by a dishonest user deliberately avoiding accountability.

## 4 Signature-in-Signature Scheme

In this section, we provide concrete constructions for SPCA and SPCA$^+$. Our starting point is the scheme of Błaśkiewicz et al. [7] (also given in App. A), which embeds an inner signature into the randomness of an outer one. Building on this idea, we generalize their concept into a generic signature-in-signature ($\mathsf{SiS}$) framework that achieves SPCA security.

The main challenge lies in integrating the inner and outer signatures in a way that preserves both correctness and security. To address this, we rely on a signature scheme with *randomness recovery*, as defined by Jaeger and Stracovsky [27], which allows to recover the signing randomness from a signature. Moreover, we introduce an enhanced variant, $\mathsf{SiS}^+$, based on a signature scheme with *public randomness recovery* [27], which allows anyone to recover the signing randomness from a signature (not only a party with $sk$). We formalize the required building blocks and prove security for both $\mathsf{SiS}$ and $\mathsf{SiS}^+$. Finally, we discuss concrete instantiations of our generic constructions and show that using Schnorr signatures as the outer scheme fails to achieve pre-compromise unforgeability.

### 4.1 Building Blocks

For the outer layer we reuse building blocks developed for anamorphic signatures [27], which we briefly recall. For the inner signature scheme we require pseudorandom signatures and exclusive ownership, apart from unforgeability.

**Signature Scheme with Randomness Recovery** A signature scheme with explicit randomness provides a signing algorithm that is deterministic once the randomness $r$ is fixed, i.e., $\mathsf{Sign}(sk, m; r)$ is deterministic. As in a standard scheme, $\mathsf{KGen}$ generates keys and $\mathsf{Vf}$ verifies signatures. In addition, we require

| **Experiment** $\mathsf{Exp}^{\mathsf{UNF-CRA}}_{\mathcal{A},\Pi}(\lambda)$ | **Experiment** $\mathsf{Exp}^{\mathsf{Rand}}_{\mathcal{A},\Pi}(\lambda)$ | **Oracle** $\mathcal{O}(m)$ |
|---|---|---|
| $(sk, pk) \leftarrow \mathsf{KGen}(1^\lambda)$ | $(sk, pk) \leftarrow \mathsf{KGen}(1^\lambda)$ | **if** $b = 0$: $\sigma := \mathsf{Sign}(sk, m)$ |
| $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}(sk,\cdot;\cdot)}(pk)$ | $b \xleftarrow{r} \{0,1\}$ | **if** $b = 1$: $\sigma \xleftarrow{r} \Sigma$ |
| **return** 1 **if** $\mathsf{Vf}(pk, m^*, \sigma^*) = 1 \ \wedge$ | $b' \leftarrow \mathcal{A}^{\mathcal{O}(\cdot)}(1^\lambda)$ | **return** $\sigma$ |
| $m^*$ was not queried to $\mathsf{Sign}(sk, \cdot; \cdot)$ | **return** 1 **if** $b = b'$ | |

Fig. 2: Security experiments for UNF-CRA and Pseudorandomness.

the randomness $r$ to be recoverable from a signature $\sigma$, which can be done with algorithm $\mathsf{ExtRand}$. Jaeger and Stracovsky [27] distinguish two variants. A signature scheme is *randomness recovering*, if extracting $r$ from $\sigma$ requires knowledge of the secret key $sk$ (as e.g., in Schnorr signatures). It is *publicly randomness recovering*, if $r$ can be extracted from the signature alone, without $sk$.

**Definition 7 ((Publicly) Randomness Recovering Signature Scheme [27]).** *A (publicly) randomness recovering signature scheme* $\Pi = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vf}, \mathsf{ExtRand})$ *for message space* $\mathcal{M}$ *and randomness space* $\mathcal{R}$ *consists of algorithms:*

$\mathsf{KGen}(1^\lambda) \to (sk, pk)$: *Outputs key pair* $(sk, pk)$.

$\mathsf{Sign}(sk, m; r) \to \sigma$: *Deterministic sign algorithm. Takes as input a secret key* $sk$*, a message* $m \in \mathcal{M}$ *and a randomness* $r \in \mathcal{R}$ *and returns a signature* $\sigma$.

$\mathsf{Vf}(pk, m, \sigma) \to 0/1$: *Returns 1 if* $\sigma$ *is a valid signature on* $m$ *under* $pk$*, else 0.*

$\mathsf{ExtRand}(pk, \boxed{sk,} m, \sigma) \to r$: *Takes public key* $pk$*, message* $m$*, and signature* $\sigma$*, and returns randomness* $r$*. If* $sk$ *is not required for extraction, then the scheme is publicly randomness recovering.*

Jaeger and Stracovsky [27] also give examples of randomness recovering schemes: Schnorr [44] and ElGamal [17] signatures are randomness recovering, while RSA-PSS [5] and Boneh-Boyen signatures [8] are publicly randomness recovering.

*UNF-CRA.* From the randomness recovering signature scheme, we require unforgeability under chosen randomness attacks (UNF-CRA) as defined by Jaeger and Stracovsky [27]. This notion strengthens standard unforgeability by demanding security even when the adversary obtains signatures on messages of its choice *under randomness of its choice*. The corresponding security experiment (Fig. 2) mirrors the classical UNF-CMA game, with the sole modification that the signing oracle accepts both a message and randomness specified by the adversary.

**Definition 8 (UNF-CRA [27]).** *A randomness recovering signature scheme* $\Pi$ *is unforgeable under chosen randomness attacks (UNF-CRA) if for every PPT adversary* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}$ *such that:* $\Pr[\mathsf{Exp}^{\mathsf{UNF-CRA}}_{\mathcal{A},\Pi}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$.

*Unique Secret Keys.* For the outer signature scheme we further require that it has unique secret keys. Therefore, we assume there is a key checking algorithm $\mathsf{PkVf}(sk, pk)$, which outputs 1 if $sk$ is a secret key corresponding to $pk$ (in discrete

logarithm based schemes where $pk = g^{sk}$, the algorithm would just check whether $pk = g^{sk}$). Then, the signature scheme has unique secret keys, if no adversary can provide two distinct $sk \neq sk'$, which belong to the same $pk$. Formally:

**Definition 9 (Unique Secret Keys).** *A signature scheme $\Pi$ has unique secret keys (USK) if there exists a negligible function* $\mathsf{negl}$ *for any PPT adversary $\mathcal{A}$ such that* $\Pr[(pk, sk, sk') \leftarrow \mathcal{A}(1^\lambda) : \mathsf{PkVf}(sk, pk) = \mathsf{PkVf}(sk', pk) = 1 \wedge sk \neq sk'] \leq \mathsf{negl}(\lambda)$

For the inner signature scheme, we define two additional security properties: pseudorandomness and exclusive ownership.

*Pseudorandom Signatures.* With pseudorandomness we refer to the fact that signatures should be indistinguishable from random elements of the signature space to anyone who does not know the $pk$ of the signature scheme. We formalize this as a security experiment, where the adversary gets an oracle producing either real signatures or random elements of the signature space and the adversary's task is to decide which is the case. If the adversary cannot do this with more than negligible probability, then the signature scheme has pseudorandom signatures.

**Definition 10 (Pseudorandom Signatures).** *A signature scheme $\Pi$ has pseudorandom signatures if there exists a negligible function* $\mathsf{negl}$ *for any PPT adversary $\mathcal{A}$ such that* $\Pr[\mathsf{Exp}^{\mathsf{Rand}}_{\mathcal{A},\Pi}(\lambda) = 1] \leq 1/2 + \mathsf{negl}(\lambda)$

*Exclusive Ownership.* This property requires that signatures commit to a single $pk$, i.e. given a public key and access to a signing oracle, the adversary should not be able to find a different $pk'$ under which an honestly generated signature verifies. This property was formalized by Cremers et al. [10] as "strong conservative exclusive ownership".

**Definition 11 (Exclusive Ownership).** *A signature scheme $\Pi$ has exclusive ownership (EO) if there exists a negligible function* $\mathsf{negl}$ *for any PPT adversary $\mathcal{A}$ such that* $\Pr[(m, \sigma, pk') \leftarrow \mathcal{A}^{\mathsf{Sign}(sk,\cdot)}(pk) : \mathsf{Vf}(pk', m, \sigma) = 1 \wedge (m, \sigma) \in Q \wedge pk \neq pk'] \leq \mathsf{negl}(\lambda)$ *where $(sk, pk) \xleftarrow{r} \mathsf{KGen}(1^\lambda)$, and $Q$ consists of pairs of $(m, \sigma)$ s.t. $m$ was queried to $\mathsf{Sign}(sk, \cdot)$ with response $\sigma$.*

### 4.2 SiS and SiS$^+$

We now define our constructions SiS and SiS$^+$. The algorithms for both are given in Fig. 3.

SiS. Our first construction, SiS, generalizes the construction from Błaśkiewicz et al. [7] using the building blocks introduced above. It uses two signature schemes: an inner signature scheme $\Pi_{\mathsf{in}}$ and an outer randomness recovering signature scheme $\Pi_{\mathsf{out}}$.

KGen outputs key pairs for both schemes, where $(sk, pk)$ are the keys of the outer scheme for security parameter $\lambda$ and $(ask, avk)$ the keys of the inner

Let $\Pi_{\mathsf{in}} = (\mathsf{KGen}_{\mathsf{in}}, \mathsf{Sign}_{\mathsf{in}}, \mathsf{Vf}_{\mathsf{in}})$ be a signature scheme with message space $\mathcal{M}$ and signature space $\Sigma_{in}$. Let $\Pi_{\mathsf{out}} = (\mathsf{KGen}_{\mathsf{out}}, \mathsf{Sign}_{\mathsf{out}}, \mathsf{Vf}_{\mathsf{out}}, \mathsf{ExtRand})$ be a signature scheme with explicit randomness, with message space $\mathcal{M}$ and randomness space $\mathcal{R}$. Let $\mathsf{MapR} : \Sigma_{in} \to \mathcal{R}$ be a bijective function. Then, $\boxed{\mathsf{SiS} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vf})}$ and $\mathsf{SiS}^+ = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vf})$ for message space $\mathcal{M}$ are defined as follows:

$\underline{\mathsf{KGen}(1^\lambda)}$: $(sk, pk) \xleftarrow{r} \mathsf{KGen}_{\mathsf{out}}(1^\lambda), (ask, avk) \xleftarrow{r} \mathsf{KGen}_{\mathsf{in}}(1^\lambda)$. Return $(sk, pk, ask, avk)$.

$\underline{\mathsf{AccSign}(sk, ask, m)}$: $\sigma_{\mathsf{in}} := \mathsf{Sign}_{\mathsf{in}}(ask, m)$, $r := \mathsf{MapR}(\sigma_{\mathsf{in}})$, $\sigma := \mathsf{Sign}_{\mathsf{out}}(sk, m; r)$. Return $\sigma$.

$\underline{\mathsf{Vf}(pk, m, \sigma)}$: Return $\mathsf{Vf}_{\mathsf{out}}(pk, m, \sigma)$.

$\underline{\mathsf{Reval}(\boxed{sk,}pk, avk, m, \sigma)}$: $r := \mathsf{ExtRand}(pk, \boxed{sk,}m, \sigma)$. $\sigma_{\mathsf{in}} := \mathsf{MapR}^{-1}(r)$.
Return 1 if $\mathsf{Vf}_{\mathsf{out}}(pk, m, \sigma) = 1 \wedge \mathsf{Vf}_{\mathsf{in}}(avk, m, \sigma_{\mathsf{in}}) = 1 \boxed{\wedge\ \mathsf{PkVf}(sk, pk) = 1}$, else return 0.

Fig. 3: The $\boxed{\mathsf{SiS}}$ and $\mathsf{SiS}^+$ scheme. $\mathsf{SiS}$ includes $\boxed{\text{highlighted text}}$

scheme for $\kappa$. To sign a message, the signer first generates an inner signature under $ask$, maps it to randomness $r$ via a transition function $\mathsf{MapR}$, and then uses $r$ in the outer signing algorithm with $sk$. Standard verification relies only on $pk$. Upon compromise, $\mathsf{Reval}$ extracts the randomness from the outer signature using $\mathsf{ExtRand}$ with $sk$, applies $\mathsf{MapR}^{-1}$ to recover the inner signature, and verifies both signatures with their respective verification algorithms. It also checks consistency between the provided $sk$ and $pk$ using $\mathsf{PkVf}$.

$\mathsf{SiS}^+$. Our second construction, $\mathsf{SiS}^+$, follows the same design as $\mathsf{SiS}$ but uses an outer scheme which is *publicly* randomness recovering. This allows $\mathsf{Reval}$ to run without $sk$, since $\mathsf{ExtRand}$ can be executed publicly to recover the randomness. Consequently, $\mathsf{SiS}^+$ is a SPCA$^+$ scheme. Because $sk$ is not involved in $\mathsf{Reval}$, the check $\mathsf{PkVf}(sk, pk)$ is also omitted from $\mathsf{Reval}$. Note that in both $\mathsf{SiS}$ and $\mathsf{SiS}^+$, the algorithm $\mathsf{AccSign}$ can be decomposed into a component that depends only on $ask$ and a subsequent component that uses $sk$. This structure is compatible with a TEE that stores only $ask$ and performs a conventional signing operation.

### 4.3 Security Analysis of SiS and SiS$^+$.

We analyze the security of $\mathsf{SiS}$ and $\mathsf{SiS}^+$ in a unified manner, as both schemes behave almost identically. We identify the properties required from the underlying signature schemes to ensure pre-compromise unforgeability, post-compromise accountability, indistinguishability, and accountable-key committing.

**Pre-Compromise Unforgeability** We first consider pre-compromise unforgeability, where the objective is to prevent forgeries that verify under the standard verification algorithm $\mathsf{Vf}$ with respect to $pk$. In this setting, the adversary has access to an $\mathsf{AccSign}$ oracle and learns all keys except the secret key $sk$.

At a high level, any valid forgery here must also be a valid signature of the outer scheme. One might therefore expect security to follow directly from the unforgeability of the outer scheme. However, standard UNF-CMA unforgeability is

insufficient here, since the adversary indirectly controls the randomness used in outer signatures: this randomness is derived from $\mathsf{Sign}_{\mathsf{in}}(ask, m)$, which depends on the adversary's chosen message $m$. Hence the adversary is strictly more powerful than in the classical chosen-message setting. Therefore, the outer scheme must still be unforgeable if the adversary controls the randomness, which is formalized as UNF-CRA. We show that UNF-CRA security of the outer scheme yields pre-compromise unforgeability.

**Theorem 1.** *If $\Pi_{\mathsf{out}}$ is an UNF-CRA signature scheme with public randomness recovery, then $\mathsf{SiS}^{(+)}$ is pre-compromise unforgeable.*

*Proof sketch.* We sketch the proof here; the full proof is given in App. B. We prove this theorem via a straightforward reduction to the UNF-CRA security of the outer scheme. We construct an adversary in the UNF-CRA experiment that simulates the $\mathsf{AccSign}$ oracle for an adversary against $\mathsf{SiS}$ using its $\mathsf{Sign}(sk, \cdot, \cdot)$ oracle, supplying $r := \mathsf{MapR}(\mathsf{Sign}_{\mathsf{in}}(ask, m))$ as the signing randomness (it can choose $(ask, avk)$ itself). If an adversary wins the pre-compromise unforgeability experiment for $\mathsf{SiS}$ by outputting a message–signature pair $(m, \sigma)$ such that $\mathsf{Vf}_{\mathsf{out}}(pk, m, \sigma) = 1$, where $m$ was never queried to the $\mathsf{AccSign}$ oracle, then this also constitutes a fresh forgery in the UNF-CRA experiment, since the freshness conditions coincide. Any successful forgery against $\mathsf{SiS}$ therefore yields a successful UNF-CRA forgery against the outer scheme.

*Is UNF-CRA too strong?* One might question whether UNF-CRA is stronger than required for our setting: the definition allows the adversary to *choose* the randomness, while in $\mathsf{SiS}$ the adversary only *influences* it indirectly by selecting $m$, which is mapped to $r = \mathsf{Sign}_{\mathsf{in}}(ask, m)$. In principle, one could formalize a weaker property tailored to this restricted influence and prove security of $\mathsf{SiS}$ and $\mathsf{SiS}^{+}$ under that assumption. However, such a property would be construction-specific, depending on both the inner and outer scheme, and thus offer limited general insight. We therefore adopt the cleaner requirement that the outer scheme be UNF-CRA-secure.

**Post-Compromise Accountability** For post-compromise accountability, the adversary knows $(sk, pk, avk)$ and its goal is to forge an accountable signature. Since the adversary already holds $sk$, it can trivially generate outer signatures. However, it cannot produce valid inner signatures if the inner scheme is unforgeable. Hence, the security of $\mathsf{SiS}^{(+)}$ in this setting reduces directly to the unforgeability of the inner scheme.

**Theorem 2.** *If $\Pi_{\mathsf{in}}$ is an UNF-CMA signature scheme, then $\mathsf{SiS}^{(+)}$ is post-compromise accountable.*

*Proof Sketch.* We sketch the proof here; the full proof is given in App. B. The argument proceeds via a straightforward reduction to the UNF-CMA security of the inner scheme. We construct an adversary in the UNF-CMA experiment

for $\Pi_{\mathsf{in}}$ that simulates the $\mathsf{AccSign}$ oracle as follows: on input a message $m$, it forwards $m$ to its own $\mathsf{Sign}$ oracle, encodes the resulting inner signature as randomness using $\mathsf{MapR}$, and then signs $m$ using this randomness and the outer signing key $sk$, which it can generate itself. This perfectly simulates the $\mathsf{AccSign}$ oracle. If the adversary against $\mathsf{SiS}$ outputs a message–signature pair $(m, \sigma)$ that verifies under $\mathsf{Reval}$ with respect to $pk$ and $avk$, then the reduction can extract the randomness from $\sigma$, which yields a valid inner-signature forgery on $m$ with respect to $avk$ and $\mathsf{Vf}_{\mathsf{in}}$. Consequently, any successful post-compromise accountability attack against $\mathsf{SiS}$ implies a successful UNF-CMA attack against $\Pi_{\mathsf{in}}$.

**Indistinguishability** For indistinguishability, we first define the reference signing algorithm $\mathsf{Sign}$ to which our scheme should be indistinguishable. In our construction, the resulting signatures are outer signatures whose randomness encodes additional structure. We require that these structured signatures be indistinguishable from signatures of the outer scheme generated using a uniformly random value as randomness. Formally, let $\mathsf{Sign}_r$ be defined as follows:

$\mathsf{Sign}_r(sk, m)$: Sample $r \xleftarrow{r} \mathcal{R}$. Return $\sigma := \mathsf{Sign}_{\mathsf{out}}(sk, m; r)$.

This definition reflects the standard behavior of most signature schemes: the randomness used in signing is chosen uniformly at random. The purpose of our construction is precisely to embed additional structure into this randomness without altering its apparent distribution.

To ensure indistinguishability, the inner scheme must produce signatures that are indistinguishable from random values to anyone who does not know the verification key $avk$. This requirement is satisfied if the inner scheme provides pseudorandom signatures. The proof of the following theorem is in App. B.

**Theorem 3.** *If $\Pi_{\mathsf{in}}$ has pseudorandom signatures, then $\mathsf{SiS}^{(+)}$ is indistinguishable with respect to $\mathsf{Sign}_r$.*

**Accountable-Key Committing** We analyze accountable-key committing separately for $\mathsf{SiS}$ and $\mathsf{SiS}^{+}$, as they depend on different properties. We start with the easier case of $\mathsf{SiS}^{+}$: Accountable-key committing requires that an honestly generated accountable signature that verifies under $avk$ cannot also verify under a different key $avk'$. Concretely, given a signature $\sigma$, the adversary should not be able to present an alternative inner verification key such that the extracted inner signature (via $\mathsf{ExtRand}$) also verifies. This property follows directly from the exclusive ownership of the inner scheme.

Accountable-key committing for $\mathsf{SiS}$ is analogous, but here $sk$ is also involved in $\mathsf{Reval}$, since $\mathsf{ExtRand}(pk, sk, \sigma)$ depends on it. Thus, an adversary could attempt to provide a different $sk$ which maps the same signature to a different randomness, and hence to a different inner signature that might verify under the adversary's $avk$. To rule this out, we additionally require that no two distinct $sk$ correspond to the same $pk$. This is why $\mathsf{Reval}$ explicitly checks that the

provided $sk$ matches $pk$. An adversary able to provide a different $sk$ for the same $pk$ would break the unique secret key property. A proof of the following theorem is given in App. B.

**Theorem 4.** *If $\Pi_{\mathsf{in}}$ has exclusive ownership then $\mathsf{SiS}^+$ is accountable-key committing. $\mathsf{SiS}$ is accountable-key committing if $\Pi_{\mathsf{in}}$ has exclusive ownership and $\Pi_{\mathsf{out}}$ unique secret keys.*

### 4.4 Insecurity of Schnorr-in-Schnorr

Błaśkiewicz et al. [7] instantiate the inner and outer scheme with Schnorr signatures. We show that this does not achieve pre-compromise unforgeability.

In the pre-compromise unforgeability experiment, the adversary is given $(ask, avk, pk)$ and must produce a signature that verifies under $pk$. Our attack exploits two properties of Schnorr signatures. First, a Schnorr signature $\sigma = (e, s)$ on $m$ satisfies $s = r + sk \cdot e$, where $r$ is the randomness used during signing. Thus, knowledge of $(m, \sigma, r)$ allows recovery of the secret key as $sk = (s - r) \cdot e^{-1}$. Second, in $\mathsf{SiS}$, the randomness used by the outer Schnorr signature is derived from an inner signature on $m$ under $ask$. To fit the outer randomness space, the inner scheme operates over a group of roughly half the size, and thus has a correspondingly smaller randomness space. Given $ask$ and an accountable signature $\sigma = (e, s)$ on a message $m$, an adversary can brute-force over this space by enumerating inner signatures with different randomness values, mapping each candidate $\sigma_{\mathsf{in}}$ to $r = \mathsf{MapR}(\sigma_{\mathsf{in}})$, and attempting key recovery via $sk = (s - r) \cdot e^{-1}$. Correctness is efficiently checked using the public key $pk$. This brute-force attack reduces security from $2^{\lambda}$ to $2^{\kappa}$; in particular, a 128-bit Schnorr instantiation yields only about 64-bit pre-compromise security. A formal description of the attack is given in App. D.

### 4.5 $\mathsf{SiS}$ and $\mathsf{SiS}^+$ Instantiation

We now discuss possible instantiations of the building blocks. A summary of the required security properties is given in Table 1 in App. B.

**Outer Scheme** The outer signature scheme must be randomness recovering, achieve UNF-CRA security and have unique secret keys. The unique secret keys property comes naturally in most discrete-logarithm or RSA-based schemes, where each public key $pk$ has a unique corresponding secret key $sk$, e.g. if $sk \in \mathbb{Z}_q$ with $g^{sk} = pk$ for a generator $g$ of a group $\mathbb{G}$ of order $q$. In contrast, some lattice-based constructions, such as ML-DSA [43] and Falcon [21], allow multiple secret keys to correspond to the same public key.

*UNF-CRA Secure Signature Schemes* Jaeger and Stracovsky [27] showed that RSA-PSS are publicly randomness recovering and UNF-CRA secure. Thus, they can be used as the outer signature scheme. Alternatively, we show how any

deterministic UNF-CMA signature scheme can be transformed into a publicly randomness recovering UNF-CRA signature scheme using a collision-resistant hash function. The transformation appends an explicit random value $r$ to each signature and signs the hash of the message together with this randomness. Thus, the randomness does not depend on the $sk$ of the outer scheme but becomes part of the signed message in the underlying inner scheme. As a result, any forgery produced by the adversary must specify both $m$ and $r$ and yield a valid signature on the pair. If the hash function is collision resistant, then $H(m, r)$ is guaranteed to be a fresh message in the underlying signature scheme, and hence constitutes a valid forgery. This construction generalizes the approach used in randomized FDH-RSA given in [19]. We give a definition and show that the scheme achieves UNF-CRA in App. C.

**Definition 12 (Publicly Randomized Signature Scheme).** *Let* $\Pi = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vf})$ *be a deterministic signature scheme with message space* $\mathcal{M}$ *and let* $H : \{0,1\}^* \to \mathcal{M}$ *be a collision-resistant hash function. The publicly randomized signature scheme* $\Pi' := (\mathsf{KGen}', \mathsf{Sign}', \mathsf{Vf}', \mathsf{ExtRand})$ *for message space* $\mathcal{M}$ *and randomness space* $\mathcal{R}$ *is defined as:*

$\mathsf{KGen}'(1^\lambda)$: $(sk, pk) \xleftarrow{r} \mathsf{KGen}(1^\lambda)$. *Return* $(sk, pk)$.
$\mathsf{Sign}'(sk, m; r)$: $\sigma' := \mathsf{Sign}(sk, H(m, r))$. *Return* $\sigma := (\sigma', r)$.
$\mathsf{Vf}'(pk, m, \sigma)$: *Parse* $\sigma := (\sigma', r)$. *Return 1 if* $\mathsf{Vf}(pk, H(m, r), \sigma') = 1$
$\mathsf{ExtRand}(pk, sk, m, \sigma)$: *Parse* $\sigma := (\sigma', r)$. *Return* $r$.

**Theorem 5.** *If* $\Pi$ *is a UNF-CMA secure signature scheme and if* $H$ *is a collision-resistant hash function, then the publicly randomized signature construction in Def. 12 is UNF-CRA secure.*

**Inner Scheme** The inner signature scheme is required to satisfy UNF-CMA security, exclusive ownership, and pseudorandomness. Schnorr signatures have been shown to achieve UNF-CMA [44,20,42] under the DL assumption in the random oracle model, and to achieve exclusive ownership [11] if $H$ is collision-resistant. We show that it also has pseudorandom signatures in the random oracle model.

**Theorem 6.** *If* $H$ *is modeled as a random oracle, the Schnorr signature scheme has pseudorandom signatures.*

*Proof.* We show that Schnorr signatures are indistinguishable from uniformly random elements of the signature space to an adversary who does not know $pk$. Recall that a Schnorr signature is $\sigma = (e, s)$, where $r \leftarrow \mathbb{Z}_q$, $e = H(m, g^r)$, and $s = r + sk \cdot e \bmod q$. For a single signature $(e, s)$, indistinguishability is immediate: $s$ is uniformly distributed in $\mathbb{Z}_q$ since $r$ is uniform, and $e$ is a random oracle output that is uniformly distributed in $\mathbb{Z}_q$. Thus $(e, s)$ is indistinguishable from a random element of $\mathbb{Z}_q \times \mathbb{Z}_q$. For multiple signatures $(e_i, s_i)$ with $e_i = H(m_i, g^{r_i})$ and $s_i = r_i + sk \cdot e_i \bmod q$, there is apparent structure, since all tuples verify under the same $pk$. To argue pseudorandomness, we must show that an adversary cannot

exploit this. The only way to detect the structure is to query the random oracle on $H(m_i, g^{r_i})$, which corresponds to querying $H(m_i, g^{s_i}/pk^{e_i})$ for an unknown $pk$. The probability that the adversary guesses such an input is therefore bounded by $q/2^{\lambda}$, where $q$ is the number of random oracle queries made by $\mathcal{A}$.

**Acknowledgements.** The work of Johannes was partially supported by the German Federal Ministry of Research, Technology and Space (BMFTR) as part of the GANGES2 project (16KIS2607K).

## References

1. Abdalla, M., Benhamouda, F., Pointcheval, D.: On the tightness of forward-secure signature reductions. Journal of Cryptology **32**(1), 84–150 (2019)
2. Abdalla, M., Reyzin, L.: A new forward-secure digital signature scheme. In: ASIACRYPT '00
3. BBC News: Project sauron malware hidden for five years (2016), `https://www.bbc.com/news/technology-37021957`, accessed: 2026-06-03
4. Bellare, M., Miner, S.K.: A forward-secure digital signature scheme. In: CRYPTO '99
5. Bellare, M., Rogaway, P.: The exact security of digital signatures-how to sign with rsa and rabin. In: EUROCRYPT '96
6. Błaśkiewicz, P., Kutyłowski, M.: Chaining electronic seals: An eidas compliant framework for controlling sscd. In: ACIIDS '22
7. Błaśkiewicz, P., Kutyłowski, M., Słowik, M.: Signature-in-signature: the last line of defence in case of signing key compromise. In: ESORICS '21
8. Boneh, D., Boyen, X.: Short signatures without random oracles. In: EUROCRYPT '04
9. Boneh, D., Drijvers, M., Neven, G.: Compact multi-signatures for smaller blockchains. In: ASIACRYPT '18
10. Cremers, C., Düzlü, S., Fiedler, R., Fischlin, M., Janson, C.: Buffing signature schemes beyond unforgeability and the case of post-quantum signatures. In: IEEE S&P '21
11. Dayanikli, D., Lehmann, A.: Password-based credentials with security against server compromise. In: ESORICS '23
12. Deo, A., Libert, B.: Anamorphic signatures with dictator and recipient unforgeability for long messages. In: ASIACRYPT '25
13. Desmedt, Y.: Threshold cryptosystems. In: AUSCRYPT '92
14. Dodis, Y., Katz, J., Xu, S., Yung, M.: Key-insulated public key cryptosystems. In: EUROCRYPT '02
15. Dodis, Y., Katz, J., Xu, S., Yung, M.: Strong key-insulated signature schemes. In: PKC '03
16. Durnoga, K., Pomykała, J., Trabszys, T.: Digital signature with secretly embedded warning. Control and Cybernetics **42**(4) (2013)
17. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE transactions on information theory **31**(4), 469–472 (1985)
18. ENISA: Operation black tulip: Certificate authorities lose authority. Tech. rep. (2011)

19. Fleischhacker, N., Günther, F., Kiefer, F., Manulis, M., Poettering, B.: Pseudorandom signatures. In: ACM ASIACCS '13
20. Fleischhacker, N., Jager, T., Schröder, D.: On tight security proofs for schnorr signatures. Journal of Cryptology **32**(2), 566–599 (2019)
21. Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z., et al.: Falcon: Fast-fourier lattice-based compact signatures over ntru. Submission to the NIST's post-quantum cryptography standardization process **36**(5), 1–75 (2018)
22. Fraser, A., Garms, L., Quaglia, E.A.: On the incoercibility of digital signatures. In: CSF '23
23. Guri, M.: Beatcoin: Leaking private keys from air-gapped cryptocurrency wallets. In: IEEE iThings/GreenCom/CPSCom/SmartData '18
24. Hülsing, A., Busold, C., Buchmann, J.: Forward secure signatures on smart cards. In: SAC '12
25. Itakura, K.: A public-key cryptosystem suitable for digital multisignature. NEC research and development **71**, 1–8 (1983)
26. Itkis, G., Reyzin, L.: Sibir: Signer-base intrusion-resilient signatures. In: CRYPTO '02
27. Jaeger, J., Stracovsky, R.: Dictators? friends? forgers.: Breaking and fixing unforgeability definitions for anamorphic signature schemes. In: ASIACRYPT '24
28. Komlo, C., Goldberg, I.: Frost: Flexible round-optimized schnorr threshold signatures. In: SAC '20
29. Kozák, K., Kwon, B.J., Kim, D., Dumitraş, T.: Issued for abuse: Measuring the underground trade in code signing certificate. arXiv:1803.02931 (2018)
30. Kozlov, A., Reyzin, L.: Forward-secure signatures with fast key update. In: SCN '02
31. Kutyłowski, M., Kubiak, P.: Lightweight digital signature with secretly embedded warning. Control and Cybernetics **42**(4), 825–827 (2013)
32. Kutyłowski, M., Lauks-Dutka, A., Błaśkiewicz, P.: Darknet signatures: public ledgers and a threat of illegal betting platforms. International Journal of Information Security **24**(3), 1–19 (2025)
33. Kutyłowski, M., Persiano, G., Phan, D.H., Yung, M., Zawada, M.: Anamorphic signatures: Secrecy from a dictator who only permits authentication! In: CRYPTO '23
34. Libert, B., Quisquater, J.J., Yung, M.: Efficient intrusion-resilient signatures without random oracles. In: Inscrypt '06
35. Liu, W., Liu, Z., Nguyen, K., Yang, G., Yu, Y.: A lattice-based key-insulated and privacy-preserving signature scheme with publicly derived public key. In: ESORICS '20
36. Liu, Z., Yang, G., Wong, D.S., Nguyen, K., Wang, H.: Key-insulated and privacy-preserving signature scheme with publicly derived public key. In: IEEE EuroS&P '19
37. Malkin, T., Obana, S., Yung, M.: The hierarchy of key evolving signatures and a characterization of proxy signatures. In: EUROCRYPT '04
38. Van der Meulen, N.: Diginotar: Dissecting the first dutch digital disaster. Journal of strategic security **6**(2), 46–58 (2013)
39. MSRC: Storm-0558 key acquisition investigation results (2023), `https://www.microsoft.com/en-us/msrc/blog/2023/09/results-of-major-technical-investigations-for-storm-0558-key-acquisition/`
40. Naccache, D., Pointcheval, D., Tymen, C.: Monotone signatures. In: FC '01
41. Nemec, M., Sys, M., Svenda, P., Klinec, D., Matyas, V.: The return of coppersmith's attack: Practical factorization of widely used rsa moduli. In: ACM CCS '17

42. Neven, G., Smart, N.P., Warinschi, B.: Hash function requirements for schnorr signatures. Journal of Mathematical Cryptology **3**(1), 69–87 (2009)
43. NIST: Module-lattice-based digital signature standard (fips 204). Federal Information Processing Standard (FIPS) 204 (2024)
44. Schnorr, C.P.: Efficient signature generation by smart cards. Journal of Cryptology **4**(3), 161–174 (1991)
45. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: FOCS '94
46. Xu, L., Chen, L., Gao, Z., Fan, X., Doan, K., Xu, S., Shi, W.: Kcrs: a blockchain-based key compromise resilient signature system. In: BlockSys 19

## A Schnorr-in-Schnorr Signatures

We describe the $\mathsf{SiS}$ instantiation with Schnorr signatures by Błaśkiewicz et al. [7]:

**Definition 13 (Schnorr-in-Schnorr Signature).** *Let $(\mathbb{G}, g, q)$ define a cyclic group $\mathbb{G}$ of prime order $q$ with generator $g$. Further let $(\mathbb{G}_{\mathsf{in}}, g_{\mathsf{in}}, p)$ define a cyclic group $\mathbb{G}_{\mathsf{in}}$ of order $p$ with generator $g_{\mathsf{in}}$ such that the bitlength of $p$ is half of $q$. Let $H_p : \{0,1\}^* \to \mathbb{Z}_p$, $H_q : \{0,1\}^* \to \mathbb{Z}_q$ be hash functions and let $\mathsf{MapR} : \mathbb{Z}_p \times \mathbb{Z}_p \to \mathbb{Z}_q$. The Schnorr-in-Schnorr scheme $\mathsf{SchiSch} = (\mathsf{KGen}, \mathsf{AccSign}, \mathsf{Vf}, \mathsf{Reval})$ with message space $\mathcal{M} = \{0,1\}^*$ and signature space $\Sigma = \mathbb{Z}_q \times \mathbb{Z}_q$ is defined as follows:*

$\mathsf{KGen}(1^\lambda)$: Choose $ask \xleftarrow{r} \mathbb{Z}_p$, $sk \xleftarrow{r} \mathbb{Z}_q$. Compute $avk := g_{\mathsf{in}}^{ask}, pk := g^{sk}$. Return $(sk, pk, ask, avk)$.

$\mathsf{AccSign}(sk, ask, m)$: 1. Compute inner Schnorr signature:
$r_{\mathsf{in}} \xleftarrow{r} \mathbb{Z}_p$, $e_{\mathsf{in}} := H_p(m, g_{\mathsf{in}}^{r_{\mathsf{in}}})$, $s_{\mathsf{in}} := r_{\mathsf{in}} + ask \cdot e_{\mathsf{in}} \bmod p$, $\sigma_{\mathsf{in}} := (e_{\mathsf{in}}, s_{\mathsf{in}})$.
2. Map inner signature to randomness space: $r := \mathsf{MapR}(\sigma_{\mathsf{in}})$.
3. Compute outer Schnorr signature using randomness $r$:
$e := H_q(m, g^r)$, $s := r + sk \cdot e \bmod q$, $\sigma := (e, s)$
4. Output $\sigma$.

$\mathsf{Vf}(pk, m, \sigma)$: Parse $\sigma := (e, s)$. Output 1 if $e = H_q(m, g^s \cdot pk^{-e})$.

$\mathsf{Reval}(sk, pk, avk, m, \sigma)$: 1. Retrieve inner and outer signature:
Parse $\sigma := (e, s)$. Compute $r := s - sk \cdot e \bmod q$, $\sigma_{\mathsf{in}} := \mathsf{MapR}^{-1}(r)$. Parse $\sigma_{\mathsf{in}} := (e_{\mathsf{in}}, s_{\mathsf{in}})$.
2. Verify both signatures:
Output 1 if $e = H_q(m, g^s \cdot pk^{-e})$ and $e_{\mathsf{in}} = H_p(m, g^{s_{\mathsf{in}}} \cdot avk^{-e_{\mathsf{in}}})$.

## B Security Proofs for SiS and SiS$^+$

In this section, we provide the security proofs for $\mathsf{SiS}$ and $\mathsf{SiS}^+$. We start with pre-compromise unforgeability, then show post-compromise accountability, accountable-key committing and indistinguishability.

### B.1 Proof of Theorem 1 (Pre-Compromise Unforgeability)

We prove by reduction. The proof is identical for $\mathsf{SiS}$ and $\mathsf{SiS}^+$, since they only differ in the handling of $\mathsf{Reval}$ which is irrelevant for pre-compromise unforgeability. Assume that a PPT adversary $\mathcal{A}$ wins the pre-compromise unforgeability game for $\mathsf{SiS}$ with non-negligible probability, and assume that $\Pi_{\mathsf{out}}$ is UNF-CRA secure. We construct an adversary $\mathcal{B}$ for the UNF-CRA experiment of the outer signature that wins with non-negligible probability.

*Construction of $\mathcal{B}$:* $\mathcal{B}$ receives $pk$ of $\Pi_{\mathsf{out}}$ as input and has access to signing oracle $\mathsf{Sign}_{\mathsf{out}}(sk, \cdot; \cdot)$. It first generates a key pair $(ask, avk)$ and passes $(pk, ask, avk)$ to $\mathcal{A}$. For queries $m$ by $\mathcal{A}$ to the $\mathsf{AccSign}$ oracle, $\mathcal{B}$ generates an inner signature $\sigma_{\mathsf{in}}$ using $sk$ and $m$. It then encodes the signature to randomness $r := \mathsf{MapR}(\sigma_{\mathsf{in}})$, queries $(m, r)$ to its own signing oracle, receives $\sigma$ and forwards it to $\mathcal{A}$. After receiving $(m^*, \sigma^*)$ from $\mathcal{A}$, $\mathcal{B}$ outputs the same $(m^*, \sigma^*)$ in the UNF-CRA experiment. Formally, $\mathcal{B}$ is defined as follows:

$\mathcal{B}^{\mathsf{Sign}_{\mathsf{out}}(sk, \cdot; \cdot)}(pk)$

---

1 : $(ask, avk) \xleftarrow{r} \mathsf{KGen}_{\mathsf{in}}(1^\lambda)$
2 : Simulate $\mathsf{AccSign}(sk, ask, \cdot)$ oracle on query $m_i$ as follows:
3 : Send $(m_i, \mathsf{MapR}(\mathsf{Sign}_{\mathsf{in}}(ask, m_i)))$ to oracle $\mathsf{Sign}_{\mathsf{out}}(sk, \cdot; \cdot)$ and receive $\sigma_i$
4 : **return** $\sigma_i$
5 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{AccSign}(sk, ask, \cdot)}(pk, ask, avk)$
6 : **return** $(m^*, \sigma^*)$

*Analysis:* For every query to $\mathsf{AccSign}$ from $\mathcal{A}$, $\mathcal{B}$ generates a signature, encodes it into randomness, makes one query to its own oracle and forwards the answer from the oracle. It finally outputs the same output as $\mathcal{A}$. Thus, if $\mathcal{A}$ is in PPT, then so is $\mathcal{B}$.

$\mathcal{B}$ perfectly simulates the $\mathsf{AccSign}$ oracle for $\mathcal{A}$. If $\mathcal{A}$ is successful in the pre-compromise unforgeability experiment, then it holds that $\mathsf{Vf}(pk, m^*, \sigma^*) = 1$, and $m^*$ was never queried to $\mathsf{AccSign}$. By construction of $\mathsf{SiS}^{(+)}$, this means that $\mathsf{Vf}_{\mathsf{out}}(pk, m^*, \sigma^*) = 1$, thus $\sigma^*$ is a valid forgery for $m^*$ with respect to $\Pi_{\mathsf{out}}$. If $\mathcal{A}$ is successful, then it never queried $m^*$ to $\mathsf{AccSign}$, but then also $\mathcal{B}$ never queried $m^*$ to its $\mathsf{Sign}_{\mathsf{out}}$ oracle, and $m^*$ is also fresh in the UNF-CRA experiment. Thus, $\mathcal{B}$ wins the UNF-CRA experiment with if $\mathcal{A}$ wins the pre-compromise unforgeability experiment. This is a contradiction to the UNF-CRA security of $\Pi_{\mathsf{out}}$, and thus no such adversary $\mathcal{A}$ can exist. This concludes the proof.

### B.2 Proof of Theorem 2 (Post-Compromise Accountability)

In the post-compromise accountability setting, the adversary knows $(sk, pk, avk)$ and has oracle access to $\mathsf{AccSign}$. Its goal is to output a valid accountable signature. We show that any such forgery can be transformed into a valid inner forgery, yielding a reduction to the $\mathsf{UNF-CMA}$ security of the inner scheme.

Assume that a PPT adversary $\mathcal{A}$ wins the post-compromise accountability experiment for $\mathsf{SiS}$ with non-negligible probability. We construct an adversary $\mathcal{B}$ for the UNF-CMA security experiment of $\Pi_{\mathsf{in}}$ that uses $\mathcal{A}$ as a subroutine.

*Construction of $\mathcal{B}$:* $\mathcal{B}$ receives the public key $pk_{\mathsf{in}}$ of the inner scheme as input and has access to signing oracle $\mathsf{Sign}_{\mathsf{in}}(sk_{\mathsf{in}}, \cdot)$. It creates a key pair $(sk, pk)$ for the outer scheme and passes $(sk, pk, avk := pk_{\mathsf{in}})$ to $\mathcal{A}$. $\mathcal{B}$ simulates the $\mathsf{AccSign}$ oracle on input $m$ from $\mathcal{A}$ as follows: it sends $m$ to its $\mathsf{Sign}_{\mathsf{in}}$ oracle, parses the answer $\sigma_{\mathsf{in}}$ as randomness $r := \mathsf{MapR}(\sigma_{\mathsf{in}})$, and generates accountable signature $\sigma := \mathsf{Sign}_{\mathsf{out}}(sk, m; r)$. It returns $\sigma$ to $\mathcal{A}$. When $\mathcal{A}$ outputs a pair $(m^*, \sigma^*)$, it first extracts the randomness from $\sigma^*$ using $\mathsf{ExtRand}$. In the proof for $\mathsf{SiS}$ it extracts it as $r := \mathsf{ExtRand}(pk, sk, m^*, \sigma^*)$, and in $\mathsf{SiS}^+$ as $r := \mathsf{ExtRand}(pk, m^*, \sigma^*)$. Since $\mathcal{B}$ generated $sk$ itself, this makes no difference in the proof. Finally, $\mathcal{B}$ reconstructs $\sigma_{\mathsf{in}}$ from the randomness $r$ as $\sigma^*_{\mathsf{in}} := \mathsf{MapR}^{-1}(r)$, and outputs $(m^*, \sigma^*_{\mathsf{in}})$ as forgery for $\Pi_{\mathsf{in}}$.

$\mathcal{B}^{\mathsf{Sign}_{\mathsf{in}}(sk_{\mathsf{in}}, \cdot)}(pk_{\mathsf{in}})$

1 : $(sk, pk) := \mathsf{KGen}_{\mathsf{out}}(1^\lambda)$
2 : Simulate oracle $\mathsf{AccSign}(sk, ask, \cdot)$ for query $m_i$:
3 : Receive $\sigma_{in,i}$ on input $m_i$ to oracle $\mathsf{Sign}_{\mathsf{in}}(sk_{\mathsf{in}}, \cdot)$
4 : return $\sigma_i := \mathsf{Sign}_{\mathsf{out}}(sk, m_i; \mathsf{MapR}(\sigma_{in,i}))$
5 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{AccSign}(sk, ask, \cdot)}((sk, pk, avk := pk_{\mathsf{in}}))$
6 : **return** $(m^*, \mathsf{MapR}^{-1}(\mathsf{ExtRand}(pk, \boxed{sk}, m^*, \sigma^*)))$

*Analysis:* $\mathcal{B}$ invokes its $\mathsf{Sign}_{\mathsf{in}}$ oracle once per $\mathsf{AccSign}$ query from $\mathcal{A}$ and performs efficient local computation. Hence $\mathcal{B}$ is a PPT adversary whenever $\mathcal{A}$ is. $\mathcal{B}$ simulates the post-compomise accountability experiment perfectly: The keys given to $\mathcal{A}$ are as constructed in the experiment for $\mathsf{SiS}^{(+)}$, and the $\mathsf{AccSign}$ oracle is answered exactly as in the experiment. If $\mathcal{A}$ is successful in the post-compromise accountability experiment, it holds that $\mathsf{Reval}(\boxed{sk}, pk, avk, m^*, \sigma^*) = 1$. By construction of $\mathsf{SiS}^{(+)}$, this means that $\mathsf{Vf}_{\mathsf{in}}(avk, m^*, \sigma^*_{\mathsf{in}}) = 1$ for $\sigma^*_{\mathsf{in}} = \mathsf{MapR}^{-1}(\mathsf{ExtRand}(pk, \boxed{sk}, m^*, \sigma^*))$. Thus, $\mathcal{B}$'s output $(m^*, \sigma^*_{\mathsf{in}})$ is a valid message-forgery pair for $\Pi_{\mathsf{in}}$. Additionally, if $\mathcal{A}$ is successful, $m^*$ was never queried to $\mathsf{AccSign}$. Since $\mathcal{B}$ forwards these queries to $\mathsf{Sign}_{\mathsf{in}}$, this means that $m^*$ was also never queried to $\mathsf{Sign}_{\mathsf{in}}$, thus $\mathcal{B}$'s forgery is also fresh. Altogether, we have constructed an adversary $\mathcal{B}$ that is successful in the UNF-CMA experiment of $\Pi_{\mathsf{in}}$, if $\mathcal{A}$ is successful in the post-compromise unforgeability experiment against $\mathsf{SiS}^{(+)}$. Thus, no such adversary $\mathcal{A}$ can exist if $\Pi_{\mathsf{in}}$ is UNF-CMA secure. This concludes the proof.

### B.3 Proof of Theorem 3 (Indistinguishability)

We show that $\mathsf{SiS}^{(+)}$ is indistinguishable with respect to $\mathsf{Sign}_r$ if $\Pi_{\mathsf{in}}$ has pseudorandom signatures. We prove by reduction. Assume that a PPT distinguisher $\mathcal{D}$

wins the indistinguishability game for $\mathsf{SiS}^{(+)}$ with non-negligible advantage. We construct a distinguisher $\mathcal{D}'$ in the pseudorandomness experiment for $\Pi_{\mathsf{in}}$ that uses $\mathcal{D}$ as a subroutine.

*Construction of $\mathcal{D}'$:* $\mathcal{D}'$ receives $1^\lambda$ as input and has access to an oracle $\mathcal{O}_{\mathsf{in}}$ which on input $m$ returns $\sigma_{\mathsf{in}}$ which is either a inner signature $\mathsf{Sign}_{\mathsf{in}}(sk, m)$, or a random element of the signature space $\Sigma$. $\mathcal{D}'$ first generates an outer key pair $(sk, pk)$ and passes them to $\mathcal{D}$. $\mathcal{D}'$ simulates the oracle $\mathcal{O}$ for $\mathcal{D}$ as follows: for each query $m$, it forwards $m$ to its own oracle $\mathcal{O}_{\mathsf{in}}$ and receives $\sigma_{\mathsf{in}}$. $\mathcal{D}$ creates randomness $r$ from $\sigma_{\mathsf{in}}$ as $r := \mathsf{MapR}(\sigma_{\mathsf{in}})$. It then uses $sk$ to generate an outer signature: $\sigma := \mathsf{Sign}_{\mathsf{out}}(sk, m; r)$ and reutrns $\sigma$ to $\mathcal{D}$. Finally, $\mathcal{D}'$ forwards the output $b$' from $\mathcal{D}$ as its own output.

$\mathcal{D}'^{\mathcal{O}_{\mathsf{in}}(\cdot)}(1^\kappa)$

1 : $(sk, pk) \leftarrow \mathsf{KGen}_{\mathsf{out}}(1^\lambda)$
2 : For every query $m_i$ from $\mathcal{D}$ to $\mathcal{O}(\cdot)$ :
3 : Send $m_i$ to $\mathcal{O}_{\mathsf{in}}(\cdot)$ and receive $\sigma_{\mathsf{in},i}$
4 : $r_i := \mathsf{MapR}(\sigma_{\mathsf{in},i})$
5 : $\sigma_i := \mathsf{Sign}_{\mathsf{out}}(sk, m_i; r_i)$
6 : Return $\sigma_i$
7 : $b' \leftarrow \mathcal{D}^{\mathcal{O}(\cdot)}(sk, pk)$
8 : **return** $b'$

*Analysis:* For every query to $\mathcal{O}$ from $\mathcal{D}$, $\mathcal{D}'$ makes one query to $\mathcal{O}_{\mathsf{in}}$, and generates a signature from the answer. It finally outputs the same bit as $\mathcal{D}$. Clearly, if $\mathcal{D}$ is in PPT, then so is $\mathcal{D}'$. We distinguish the two cases for $\mathcal{O}_{\mathsf{in}}$:

**(I)** If $\mathcal{O}_{\mathsf{in}}(\cdot) = \mathsf{Sign}_{\mathsf{in}}(ask, \cdot)$, then for every query $m$ from $\mathcal{D}$, $\mathcal{D}'$ returns $\sigma = \mathsf{Sign}_{\mathsf{out}}(sk, m; \mathsf{MapR}(\mathsf{Sign}_{\mathsf{in}}(ask, m)))$ which is exactly what the $\mathsf{AccSign}$ oracle would return for $\mathsf{SiS}^{(+)}$. Thus, it holds that

$$\Pr[\mathcal{D}'^{\mathcal{O}_{\mathsf{in}}(\cdot)}(1^\kappa) = 1 | b = 0] = \Pr[\mathcal{D}^{\mathcal{O}(\cdot)}(1^\kappa) = 1 | b = 0]$$

**(II)** If $\mathcal{O}_{\mathsf{in}}(\cdot)$ returns $\sigma_{\mathsf{in}} \xleftarrow{r} \Sigma$, then for every query $m$ from $\mathcal{D}$, $\mathcal{D}'$ returns $\sigma = \mathsf{Sign}_{\mathsf{out}}(sk, m; \mathsf{MapR}(\sigma_{\mathsf{in}}))$. Since $\mathsf{MapR}$ is a bijection from $\Sigma$ to $\mathcal{R}$, a uniformly random $\sigma_{\mathsf{in}} \in \Sigma$ is mapped to a uniformly random element of $\mathcal{R}$. Therefore, $\mathcal{D}'$ outputs $\sigma = \mathsf{Sign}_{\mathsf{out}}(sk, m; r)$ for $r \xleftarrow{r} \mathcal{R}$, which is exactly the output of the reference signing algorithm $\mathsf{Sign}_r$ for input $m$. Thus,

$$\Pr[\mathcal{D}'^{\mathcal{O}_{\mathsf{in}}(\cdot)}(1^\kappa) = 1 | b = 1] = \Pr[\mathcal{D}^{\mathcal{O}(\cdot)}(1^\kappa) = 1 | b = 1]$$

Combining cases (I) and (II), we conclude that $\mathcal{D}'$ wins the pseudorandomness experiment for $\Pi_{\mathsf{in}}$ with the same probability that $\mathcal{D}$ wins the indistinguishability experiment for $\mathsf{SiS}^{(+)}$. Hence, if $\Pi_{\mathsf{in}}$ is pseudorandom, no such distinguisher $\mathcal{D}$ can exist, and $\mathsf{SiS}^{(+)}$ is indistinguishable.

**Proof of Theorem 4 (Accountable-Key Committing)** We start with $\mathsf{SiS}$. We show that $\mathsf{SiS}$ is accountable-key committing, when the inner signature has exclusive ownership and the outer signature has unique secret keys. We prove the theorem by reduction. Assume that a PPT adversary $\mathcal{A}$ wins the accountable-key committing game for $\mathsf{SiS}$ with non-negligible probability. Then we can construct an adversary $\mathcal{B}$ that either breaks exclusive ownership of the inner signature or unique secret keys of the outer signature.

*Construction of $\mathcal{B}$:* Let $\mathcal{B}$ be an adversary trying to break exclusive ownership, i.e. it receives a public key $pk_{\mathsf{in}}$ as input and can query a $\mathsf{Sign}_{\mathsf{in}}(sk_{\mathsf{in}}, \cdot)$ oracle. First it generates an outer key pair $(sk, pk)$ and forwards $pk$ to $\mathcal{A}$. For queries by $\mathcal{A}$, $\mathcal{B}$ emulates the $\mathsf{AccSign}$ and $\mathsf{Comp}$ oracle. For queries $m$ to $\mathsf{AccSign}$, it generates an inner signature using its sign oracle. It encodes the resulting signature $\sigma_{\mathsf{in}}$ into $r := \mathsf{MapR}(\sigma_{\mathsf{in}})$ and generates an outer signature as $\mathsf{Sign}_{\mathsf{out}}(sk, m; r)$. All signatures are stored in a list while the compromise bit is not set. The query to $\mathsf{Comp}$ is handled by returning $(sk, avk := pk_{\mathsf{in}})$ and setting the compromise bit to 1. After receiving $(m^*, \sigma^*, sk^*, avk^*)$ from $\mathcal{A}$, $\mathcal{B}$ checks if $sk \neq sk^*$. In the case of different secret keys, $\mathcal{B}$ outputs $(pk, sk, sk*)$ which breaks the unique secret keys property of $\Pi_{\mathsf{out}}$. Otherwise $\mathcal{B}$ outputs $(m^*, \mathsf{MapR}^{-1}(\mathsf{ExtRand}(pk, sk, m^*, \sigma^*)), avk, avk^*)$. Formally, we construct $\mathcal{B}$ as follows:

$\mathcal{B}^{\mathsf{Sign}_{\mathsf{in}}(ask,\cdot)}(avk)$

---

$(sk, pk) := \mathsf{KGen}_{\mathsf{out}}(1^\lambda)$
Simulate oracle $\mathsf{AccSign}(sk, ask, \cdot)$ for query $m_i$ as follows:
  Receive $\sigma_{in,i}$ on input $m_i$ to oracle $\mathsf{Sign}_{\mathsf{in}}(ask, \cdot)$
  return $\sigma_i := \mathsf{Sign}_{\mathsf{out}}(sk, m_i; \mathsf{MapR}(\sigma_{in,i}))$
Simulate $\mathsf{Comp}()$ oracl:
  $\mathsf{comp} := 1$
  return $(sk, avk)$
$(m^*, \sigma^*, sk^*, avk^*) \leftarrow \mathcal{A}^{\mathsf{AccSign}(sk,ask,\cdot),\mathsf{Comp}()}(pk)$
if $sk \neq sk^*$ then:
  **return** $(pk, sk, sk^*)$
$\sigma^*_{\mathsf{in}} := \mathsf{MapR}^{-1}(\mathsf{ExtRand}(pk, sk, m^*, \sigma^*))$
**return** $(m^*, \sigma^*_{\mathsf{in}}, avk^*)$

*Analysis:* $\mathcal{B}$ runs $\mathcal{A}$ and answers all its queries using efficient local computations. Hence, $\mathcal{B}$ is PPT whenever $\mathcal{A}$ is. Moreover, $\mathcal{B}$ perfectly simulates the $\mathsf{AccSign}$ and $\mathsf{Comp}$ oracles for $\mathcal{A}$. If $\mathcal{A}$ is a successful adversary in the accountable-key committing experiment, then for its output $(m^*, \sigma^*, sk^*, avk^*)$ it holds that $\mathsf{Reval}(sk^*, pk, avk^*, m^*, \sigma^*) = 1$, $\mathsf{Reval}(sk, pk, avk, m^*, \sigma^*) = 1$, and additionally $(sk^*, avk^*) \neq (sk, avk)$. We distinguish two cases:

(I) **Case** $sk \neq sk^*$: In this case, $\mathcal{B}$ is able to break unique secret keys of the outer scheme by outputting $(pk, sk, sk^*)$. By construction of $\mathsf{SiS}$, it must

Table 1: Overview of the necessary security properties of building blocks for $\mathsf{SiS}$ and $\mathsf{SiS}^+$.

| **Security Property** | **Requirements** | | | |
|---|---|---|---|---|
| | $\mathsf{SiS}$ | | $\mathsf{SiS}^+$ | |
| | $\Pi_{\mathsf{in}}$ | $\Pi_{\mathsf{out}}$ | $\Pi_{\mathsf{in}}$ | $\Pi_{\mathsf{out}}$ |
| Pre-Compromise Unforgeability | - | UNF-CRA | - | UNF-CRA |
| Post-Compromise Accountability | UNF-CMA | - | UNF-CMA | - |
| Accountable-Key Committing | EO | USK | EO | - |
| Indistinguishability | PR | - | PR | - |

hold that $\mathsf{Pkcheck}(sk, pk) = 1$ and $\mathsf{Pkcheck}(sk^*, pk) = 1$ as this is checked within $\mathsf{Reval}$.

(II) **Case** $sk = sk^*$: In this case, it must hold that $avk \neq avk^*$. The winning condition of $\mathcal{A}$ implies that $\mathsf{Vf}_{\mathsf{in}}(avk, m, \sigma^*_{\mathsf{in}}) = 1$ and $\mathsf{Vf}_{\mathsf{in}}(avk^*, m, \sigma^*_{\mathsf{in}}) = 1$ for $\sigma^*_{\mathsf{in}} := \mathsf{MapR}^{-1}(\mathsf{ExtRand}(pk, sk, m^*, \sigma^*))$ by construction of $\mathsf{SiS}$. Thus, $\mathcal{B}$ breaks exclusive ownership of the inner signature by outputting $(m, \sigma^*_{\mathsf{in}}, avk^*)$

We have thus constructed a successful efficient adversary $\mathcal{B}$ breaking either exclusive ownership of the inner scheme or unique secret keys of the outer scheme. Consequently, no efficient adversary $\mathcal{A}$ against accountable-key committing can exist if $\Pi_{\mathsf{in}}$ satisfies exclusive ownership and $\Pi_{\mathsf{out}}$ satisfies unique secret keys. It follows that $\mathsf{SiS}$ is accountable-key committing under these assumptions.

The accountable-key committing proof for $\mathsf{SiS}^+$ is analogous. Assume an adversary $\mathcal{A}$ breaks accountable-key committing of $\mathsf{SiS}^+$. Then, we construct $\mathcal{B}$ that breaks exclusive ownership of the inner scheme. The simulation of signing and compromise oracles proceeds as in the proof for $\mathsf{SiS}$. Given the output $(m^*, \sigma^*, avk^*)$ from $\mathcal{A}$, $\mathcal{B}$ extracts the embedded inner signature as $\sigma^*_{\mathsf{in}} := \mathsf{MapR}(\mathsf{ExtRand}(pk, m^*, \sigma^*))$ and outputs $(m^*, \sigma^*_{\mathsf{in}}, avk^*)$. By the winning condition of $\mathcal{A}$, it holds that $\mathsf{Vf}_{\mathsf{in}}(avk, m^*, \sigma^*_{\mathsf{in}}) = 1$ and $\mathsf{Vf}_{\mathsf{in}}(avk^*, m^*, \sigma^*_{\mathsf{in}}) = 1$. Thus, $\mathcal{B}$ breaks exclusive ownership of $\Pi_{\mathsf{in}}$ if $\mathcal{A}$ is a successful PPT adversary in the accountable-key committing experiment for $\mathsf{SiS}^+$. Hence, exclusive ownership of $\Pi_{\mathsf{in}}$ implies accountable-key committing of $\mathsf{SiS}^+$.

## C Proof of Theorem 5

In this section, we prove that the publicly randomized scheme of Definition 12 achieves UNF-CRA security.

*Proof.* We prove by reduction. Assume that a PPT adversary $\mathcal{A}$ wins the UNF-CRA game for hash-based randomness signatures with non-negligible probability. We will use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ that either breaks UNF-CMA security of $\Pi$ or collision-resistance of $H$. The idea is that a valid forgery $\sigma := (\sigma', r)$ for $m$ in the randomized scheme consists of a signature $\sigma'$ on $H(m, r)$ which is either a fresh forgery in the underlying signature scheme, or a collision in $H$.

*Construction of $\mathcal{B}$:* $\mathcal{B}$ receives the public key of the signature $pk$ as input. To simulate $\mathcal{A}$'s signing requests on $(m_i, r_i)$, $\mathcal{B}$ queries its own signing oracle $\mathsf{Sign}(sk, \cdot)$ on $H(m_i, r_i)$ and returns $\sigma_i = (\sigma', r_i)$ to $\mathcal{A}$. When $\mathcal{A}$ outputs a pair $(m^*, \sigma^* = (\sigma^{**}, r^*))$, $\mathcal{B}$ outputs $\sigma'^*$ as forgery on $H(m^*, r^*)$.

$\mathcal{B}^{\mathsf{Sign}(sk,\cdot)}(pk)$

1 : On query $(m_i, r_i)$ from $\mathcal{A}$ to $\mathsf{Sign}'(sk, \cdot; \cdot)$ :
2 : Receive $\sigma_i$ from query $H(m_i, r_i)$ to $\mathsf{Sign}(sk, \cdot)$
3 : Return $\sigma_i := (\sigma, r_i)$
4 : On output $(m^*, \sigma^*)$ from $\mathcal{A}$ :
5 : $\sigma^{**} := (\sigma^*, r^*)$
6 : Return $(H(m^* \| r^*), \sigma^{**})$

*Analysis:* For every query to $\mathsf{Sign}$ from $\mathcal{A}$, $\mathcal{B}$ makes one query to its own oracle and returns the answer to $\mathcal{A}$. If $\mathcal{A}$ is a PPT adversary, then so is $\mathcal{B}$. If $\mathcal{A}$ wins the UNF-CRA experiment, then it holds that $\mathsf{Vf}'(pk, m^*, \sigma^*) = 1$, and thus, $\mathsf{Vf}(pk, H(m^*, r^*), \sigma^{**}) = 1$. Thus. $\sigma^{**}$ is a valid forgery in the UNF-CMA game. We still need to show that $H(m^*, r^*)$ was not requested to the signing oracle of $\mathcal{B}$. There are two cases: (1) $\mathcal{B}$ queried $H(m^*, r^*)$ for query $(m^*, r^*)$ from $\mathcal{A}$. In this case, $\mathcal{A}$ would not win the UNF-CRA, as the forgery would not be on a fresh message. (2) It holds that $H(m^*, r^*) = H(m_i, r_i)$ for some previous query $(m_i, r_i)$ to its signing oracle. In this case, $\mathcal{B}$ can output $((m^*, r^*), (m_i, r_i))$ as a collision of $H$. Thus, this only happens with negligible probability if $H$ is collision-resistant. This implies that $\mathcal{B}$ either breaks unforgeability of $\Pi$ or collision-resistance of $H$ if there is a successful attacker $\mathcal{A}$ against UNF-CRA of $\Pi'$. Thus, if $H$ is collision resistant, and if $\Pi$ is UNF-CMA, no such adversary $\mathcal{A}$ can exist and $\Pi'$ is UNF-CRA secure.

## D Formal Attack on Schnorr-in-Schnorr

We provide a formal attack showing that the Schnorr-in-Schnorr scheme does not achieve pre-compromise unforgeability. In particular, the following attacker runs in PPT and has a success probability of $t/2^\kappa$ for a fixed $t$, which might be non-negligible in $\lambda$.

$\mathcal{A}^{\mathsf{AccSign}(sk,ask,\cdot)}(pk, ask, avk)$
Pick $m \xleftarrow{r} \mathcal{M}$, send $m$ to $\mathsf{AccSign}(sk, ask, \cdot)$ oracle and receive $\sigma = (e, s)$.
For $i = 1, \ldots, t$:
Run inner Schnorr signing algorithm on $m$ using $ask$ and fresh randomness $r_{\mathsf{in}} \xleftarrow{r} \mathbb{Z}_p$:
$e_{\mathsf{in}} := H_p(m, g^{r_{\mathsf{in}}})$, $s_{\mathsf{in}} := r_{\mathsf{in}} + ask \cdot e_{\mathsf{in}}$, $\sigma_{\mathsf{in}} := (e_{\mathsf{in}}, s_{\mathsf{in}})$
Compute outer randomness $r := \mathsf{MapR}(\sigma_{\mathsf{in}})$
Recover potential $sk$: $sk^* := (s - r) \cdot e^{-1}$
If $pk = g^{sk^*}$, break
Pick $m^* \xleftarrow{r} \mathcal{M}$, return $(m^*, \mathsf{Sign}(sk^*, ask, m^*))$.