Almost Scriptless Adaptor Signatures from any Signature Scheme

Emanuele Giunta*, Kristina Hostáková

{emanuele.giunta,kristina.hostakova}@inf.ethz.ch
ETH Zurich, Switzerland.

Abstract. An adaptor signature (AS) scheme extends a digital signature by tying together message authentication with the exchange of a secret. Such atomic exchange of a secret for a signature has key applications in blockchain protocols, e.g., payment channels, atomic swaps or coin-mixing. To ensure compatibility with the environments, AS schemes should be *scriptless* – that is, their signatures should be verifiable using the same procedure used to validate transactions' signatures, without auxiliary logic or scripts.

As shown by Erwig et al. (PKC'21), it is impossible to construct scriptless AS for standard signature schemes with unique signatures. In contrast, Liu et al. (ASIACRYPT'24) demonstrated that dropping the scriptless requirement entirely makes AS construction for any EUF-CMA secure signature scheme straightforward. This raises a fundamental question: must we abandon scriptlessness entirely to construct AS for any signature scheme?

We answer this question in the negative by introducing the notion of *almost scriptless* AS. Here, the only relaxation is that the signature is applied to an extended message – specifically, the concatenation of the original message and a random string (i.e., signing $m\|r$ instead of m). We argue that this mild extension preserves the core benefits of scriptlessness in blockchain applications.

We present a generic compiler that transforms any *functional signature* scheme into an AS, almost preserving the signature verification. In turn, show how indistinguishability obfuscation (in the CRS model) or witness encryption (in the ROM) allow turning any signature into a functional signature in a scriptless fashion. Both constructions ultimately yield almost scriptless AS for unique-witness (hard) relations and for any base signature.

1 Introduction

Adaptor signature schemes (AS for short) extend standard digital signature allowing one to embed a witness for an NP-statement into a signature, which could later be extracted once the signature is published. In addition to the key generation, signing and verification (which we call the *base signature* of an AS), AS

* Part of the work was done while the author was at IMDEA Software Institute, Madrid, Spain.

schemes offer an alternative procedure to sign (cf. Fig. 1): ① a pre-signing algorithm $\mathsf{AS.pSign}$ that pre-signs a message m with respect to a NP-statement x; ② a pre-signature verification algorithm $\mathsf{AS.pVfy}$ verifying pre-signature's validity; ③ an adaption algorithm $\mathsf{AS.Adapt}$ that using a witness w for x, transforms a pre-signature into a valid signature (sometimes called the *adapted signature*); ④ an extraction algorithm $\mathsf{AS.Ext}$ that given a pre-signature and the adapted signature extracts a (potentially different) valid witness w' for x.

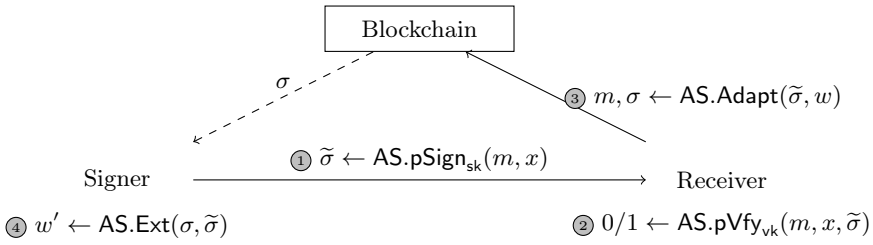

Fig. 1. Typical interaction between a Signer and a Receiver using an adaptor signature scheme. $(\mathsf{sk}, \mathsf{vk})$ is the signer's key pair from a hard relation $\mathcal{R}$, m the message being signed and (x, w) a statement witness pair, where w is known by the Receiver.

The idea of adaptor signatures (a.k.a. scriptless scripts) was put forward by Andrew Poelstra [Poe17] in the context of conditional payments on blockchains. Every blockchain supports transaction authentication, i.e., verification of a signature on a payment transaction. When designing applications on top of a blockchain, it is very convenient if additional checks are supported (typically referred to as smart contracts or scripts). However, running such scripts is very costly as they need to be downloaded and validated by all full nodes in the network. Moreover, their details are fully visible and stay on the blockchain forever. Adaptor signatures provide *an alternative to scripts* by implementing a fair purchase of a secret value over blockchain – a very important building block for many applications, including atomic swaps [Poe18, TMM22, TZY22], scalability solutions like payment channels [AEE+21, AME+21, EEE20, MMS+19], privacy enhancements [GMM+22, QPM+23], oracle-based payments [MTV+23].

Adaptor signatures were first formalized by Aumayer et al. [AEE+21] and have received increasing attention since then. Aumayer et al. proved that AS can be built with respect to Schnorr Signatures and ECDSA, the most relevant signature schemes in the blockchain context at the time. Their result was followed up by a number of works constructing AS for other concrete signatures, e.g. [EEE20, TMM21, TZY22, KH22].

Another line of works focused on generalizing the results of Aumayer et al. and presenting generic transformations from certain classes of signature schemes to AS. Erwig et al. [EFH+21] showed that ID-scheme-based signatures satisfying certain homomorphic properties can be turned into AS. More recently, Gerhart et al. [GSST24] defined a signature notion called dichotomic signatures and

showed how to construct AS from them. Both works, however, impose constraints on the underlying signature scheme. A natural question is whether AS can be constructed for *any* signature scheme.

Impossibility of generic AS. Erwig et al. [EFH+21] showed that no *unique* signature scheme (i.e., for a given message m and a verification key vk, there is only one valid signature σ) can be turned into an AS for a hard relation $\mathcal{R}$. An example of a unique signature scheme is BLS which otherwise has many blockchain friendly properties. The impossibility result is not difficult to see so let us briefly sketch it here. If there would be an AS w.r.t. a unique signature scheme and a hard relation $\mathcal{R}$, then a signer could break the hardness of $\mathcal{R}$. Consider a statement x, and a message m. Let $\widetilde{\sigma}$ be a pre-signature for m w.r.t. x, and let σ *the unique valid* signature for m. Correctness of AS says that adapting $\widetilde{\sigma}$ with a witness for x must result in a valid signature for m – hence must be equal to σ. But knowing both the pre-signature and adapted signature allows extraction of a witness for x thereby breaking the hardness of $\mathcal{R}$.

More generally, for a signer that knows only a statement x but no witness for x, it must be hard to produce a valid pre-signature w.r.t. x, and a corresponding adapted signature.

Circumventing the impossibility. A crucial condition in the AS definition (and hence the assumptions of the impossibility result) is that the verification algorithm used to verify an adapted signature has to be *the same* as the verification algorithm of the base signature scheme. And in fact, if we follow the motivation of AS, the base signature is dictated by the respective blockchain (i.e., the base signature should be the signature scheme used for transaction authorization). This property is called *scritplessness* of AS.

As demonstrated by Dai et al. [DOY22], *extending* the verification algorithm, i.e., dropping the scriptlessness requirement, suffices to circumvent the impossibility. In a nutshell, their generic transformation works as follows: to pre-sign a message m w.r.t. a statement x, the signer outputs a pre-signature $\widetilde{\sigma} = (\mathsf{Sign}(\mathsf{sk}, m\|x), x)$. To adapt this pre-signature, one only attaches a witness w for x, i.e., the adapted signature is $\sigma = (\mathsf{Sign}(\mathsf{sk}, m\|x), x, w)$. The verification algorithm then verifies the validity of the signature on $m\|x$, and checks that $(x, w) \in R$. Recently Lui et al. [LIZ24] improved this construction by hiding the witness in the adapted signature. Verification is modified in a similar way.

There are two reasons why the above transformation violates the impossibility result's assumptions.

1. *Message space extension:* The message being signed is $m\|x$ and not m.
2. *Additional checks:* The verification additionally tests $(x, w) \in R$.

From a cryptographical perspective, these might be seen as minor modifications. Application-wise, however, they introduce significant complications.

The first weakening of the AS notion is easier to justify for fair purchase over a blockchain. In this context, signing $m\|x$ instead of m only requires that additional data can be attached to a blockchain transaction. This is possible on many

blockchains, including Ethereum and Bitcoin[1], although it makes the transaction distinguishable from a simple payment transaction and slightly increases the costs. The second point is significantly more problematic. Indeed it either requires to change the native signature scheme used to validate transactions (a major and costly event) or to add the check for (x, w) *on top of* signature verification, essentially relying again on a script! We stress here that avoiding scripts is arguably the main reason to develop AS in the first place.

Given the state of the art, a natural open question is whether both weakening the AS notion are necessary to overcome the impossibility result. In particular, can we construct AS from any signature scheme if we allow for message space extension, but *no further modifications* to the underlying verification procedure? We will refer to schemes satisfying this relaxed scriptlessenes notion as *almost scriptless*.

1.1 Our Contributions

In this work, we present the first two generic AS constructions from *any* digital signature scheme without adding checks to the verification. Specifically, to overcome the impossibility in [EFH$^+$21] and in line with previous works [DOY22, LIZ24], we augment the message being signed with a random string. That is, we assume that upon signing m, a signature (with the base scheme) for $m\|r$ is actually produced, where r is a random string[2]. However, contrary to previous works, our generic transformation is almost scriptless as it keeps the underlying verification procedure unchanged.

Overall, our main contributions consist of the following three constructions.

1. A compiler of almost scriptless AS from any scriptless (or even almost scriptless) Functional Signature (FS) [BGI14] and a non-interactive zero-knowledge (NIZK), with respect to any NP-relation also supported by the FS scheme.
2. A scriptless realization of FS from any signature scheme and indistinguishability obfuscation (iO) [BGI$^+$01] supporting predicates with unique witnesses[3] in the CRS model.
3. A scriptless instantiation of FS from any signature, relying on witness encryption (WE) [GGSW13] and garbled circuits [Yao82]. The resulting scheme supports predicates with unique witness in the CRS model (for a mildly restricted class of predicates), or in the Random Oracle/Generic Group model (with no restriction beyond unique witness).

As an additional contribution, we also provide the first formalization of the *scriptlessness* property and its fine-grain variants.

[1] For Bitcoin, this can be done through the OP_RETURN opcode [OPR].

[2] As argued above, this in practice means that r is attached as metadata to the transaction m.

[3] Such limitation can be lifted in the iO case to $\mathsf{poly}(\lambda)$ many witnesses, although with some technical caveats as discussed in Section 5.1

Let us stress that the efficiency of the resulting adaptor signature schemes cannot compete with adaptor signature schemes built for concrete signature schemes (e.g., Schnorr-based AS). The goal of this work is to show feasibility of almost scriptless AS from *any* signature scheme, thereby understand the theoretical limits of the primitive better, and hopefully pave the path for more efficient AS construction for, e.g., BLS.

1.2 Technical Overview

We start recalling informally the three properties of AS schemes:

- *Pre-signature adaptability:* Adapting correctly a valid pre-signature on a message m must results in a valid signature on m.
- *Witness extractability:* No new signature σ can be computed by the receiver such that no witness is extractable from it.
- *Unlinkability:* without statement, witness and pre-signature, adapted and regular signatures are indistinguishable.

Let us remark that unforgeability of AS, a property one would expect for any signature, is formally implied by witness extractability as shown by [DOY22].

AS from Functional Signature. The first step toward our generic constructions is to realize AS from any *functional* signature scheme Σ. Informally, a FS scheme (for predicates) allows a signer holding a master secret key sk to issue a *functional* secret key sk_p for a given predicate p. Given sk_p, it is possible to sign messages satisfying p and nothing more.

The core idea of our construction is to interpret pre-signatures as functional keys. More precisely, we set the pre-signature on a message m^* w.r.t. a statement x to be a functional key sk_p for the predicate $p(m\|\alpha) : (m = m^*) \wedge (x, \alpha) \in \mathcal{R}$. A receiver holding sk_p can then sign any message $M = (m\|\alpha)$ whose payload m is the specified message m^* and random coins α are a witness for x. The FS unforgeability guarantees these are also the only messages that can be signed, implying witness extractability. In order to allow the receiver to verify the correctness of the pre-signature, we furthermore attach a NIZK, certifying that sk_p was computed correctly. A NIZK needs a crs, which we generate in a separate $\mathsf{AS.Setup}$ algorithm (and hence not the key generation algorithm $\mathsf{AS.Gen}$). The reason is that we do not want to have the crs being part of the verification key as this would imply a change in the underlying signature scheme which is undesirable in our setting. The crs is then provided as input to the pre-signing/verifying algorithms $\mathsf{AS.pSign}$ and $\mathsf{AS.pVfy}$, but the remaining algorithms are independent of the crs.

Note that the aforementioned construction would trivially leak the witness. We avoid this by masking the witness as follows: During pre-signing, the signer additionally samples a uniform string ρ and defines the predicate p to be

$$p(m||\alpha) : (m = m^*) \wedge (x, \alpha \oplus \rho) \in \mathcal{R}.$$

The masking value ρ is then added to the pre-signature. In order to turn the pre-signature into a valid signature using a witness w, the receiver uses sk_p to sign the message $M = (m^*||\alpha)$, where $\alpha = w \oplus \rho$. A signer that sees the adapted signature on M can then extract the witness w as $\alpha \oplus \rho$.

The question then is how to construct FS from any signature scheme without changing the verification procedure.

Functional Signatures from iO. Let us start by presenting a simplified version of our construction, and only later show why it fails and how to fix it. We start with any signature scheme Σ and any family of predicates $\mathcal{P}$. Our FS uses the same sk and vk of the underlying scheme as master secret key and verification key. A functional key sk_p is just going to be an *obfuscation* of $C_{\mathsf{sk},p}$ which on input a message m outputs $\mathsf{S.Sign}(\mathsf{sk}, m)$[4] if $p(m) = 1$ and $\perp$ otherwise.

Unfortunately such a simple approach fails, to the best of our knowledge, to achieve provable security. At a high level, when reducing FS unforgeability to the base signature's security, we need a way to simulate the obfuscation of $C_{\mathsf{sk},p}$ without sk. As we are not imposing any restriction on the underlying scheme, the only way for our reduction to obtain signatures is through oracle queries. We therefore face two issues:

1. All the signatures $C_{\mathsf{sk},p}$ is expected to compute, should be *programmed* in the simulated circuit. This means that information theoretically we cannot support more signatures than the *length* of $C_{\mathsf{sk},p}$.[5]
2. Our reduction needs to know essentially all messages m satisfying p *before* returning the simulated circuit, in order to embed the respective signatures.

Note that, although we do not formalize and prove this, the above issues seems to affect essentially any generic (almost scriptless) FS, when unforgeability is reduced to that of the underlying signature. Analogous limitations are discussed for the case of AS in [AEE+21, §5.1] and [GSST24, p.5]. To address such obstacles we impose restrictions on the supported class of predicates. Precisely, we assume that (i) each $p \in \mathcal{P}$ admits a *unique witness* m such that $p(m) = 1$, and (ii) that the reduction can extract m from p.[6]

The second point specifically can be achieved in a variety of ways. One is to require that p contains an extractable commitment to m. Conversely we could assume a *straight-line extractable* proof of knowledge for m is attached to p (an approach also taken by [AEE+21] for ECDSA and [TMM21] for CSI-FiSh). This could either be done in the CRS model, or in the ROM through Fischlin transform [Fis05]. The latter case however requires some care to avoid obfuscating

[4] At this stage, let us assume $\mathsf{S.Sign}$ to be deterministic – random coins generation could however be handled with standard techniques from puncturable PRF.

[5] Compressing the representation of many signatures could work for specific underlying schemes. However devising a way to do so *for any* signature scheme appears unlikely.

[6] In fact, we could generalize our construction to predicates with $\mathsf{poly}(\lambda)$ many witnesses as long as it is possible to extract all the possible witnesses from the predicate.

the random oracle[7]. Finally, specific relations, such as the preimage relation for trapdoor permutations readily allow for extraction without any modification.

Functional Signature from WE and Garbling. Our second FS scheme relies on simpler assumptions compared to iO. Our approach to remove obfuscation will be to rely on non-interactive two party computation techniques. At a high level, when asked to generate sk_p, we aim to provide the receiver with enough information to compute a signature for the only m satisfying p.

Attempting to use 2PC techniques however comes with at least two technical challenges. First, the setting is inherently highly non-interactive, allowing for only one message being sent by the signer and *no preprocessing*. Second, even if we were to allow for more interaction, the receiver may not know *yet* a satisfying assignment for p. Zooming out to the resulting AS, this is equivalent to a receiver not knowing a witness when receiving and verifying a pre-signature.[8]

We solve both issues by combining garbling with witness encryption (WE) for NP. Briefly, a WE scheme for an NP-relation $\mathcal{R}$ allows encrypting a message m w.r.t. a statement x such that it can be decrypted using any witness for x. A (projective) garbling of a boolean circuit C is an encoding of C assigning two labels to each wire (corresponding to 0 or 1 state). Evaluating a garbling C on x requires the input labels encoding x.

Up to mild simplifications, our construction generates functional keys sk_p as follows. Initially the circuit $\mathsf{S.Sign}(\mathsf{sk}, \cdot)$ is garbled, obtaining input labels $X_{1,0}, X_{1,1}, \ldots, X_{n,0}, X_{n,1}$. Ideally, we wish to set sk_p as the garbling above, plus the labels $X_{1,m_1}, \ldots, X_{n,m_n}$ encoding the *only* satisfying assignment m for p. However, such m is not known to the signer. Addressing this is where WE comes into play. The signer indeed proceed to encrypt $X_{i,b}$ with respect to the statement "there exists m such that $p(m) = 1$ and $b = m_i$". WE combined with our restriction on unique witnesses then suffices to guarantee that the receiver can successfully decrypt *only* labels encoding m such that $p(m) = 1$.

Removing the need of extractable predicates. As in the case of our iO construction, this second approach still requires *the reduction* to know the witness m *before* returning sk_p. Specifically, this is necessary to correctly program a signature σ for m into a simulated garbling.

This could be achieved again through extractable predicates. Contrary to the construction from obfuscation, a different trade-off is possible. We can indeed remove any extractability assumption on $\mathcal{P}$ (though we still need unique witnesses), relying instead on the Random Oracle Model. Thanks to the ROM a lazy simulation strategy becomes viable, where σ is injected during an evaluation of sk_p, and not when sk_p itself is computed. To do so we need two mechanisms in place: one to *observe* a satisfying m for p once an adversary successfully starts the computation of σ with sk_p, and one to *program* the desired signature once m is known.

[7] See [HJK+16] for a discussion on why obfuscation of RO aided circuits is unsound.

[8] Indeed in many AS applications, such as payment channels or atomic swaps, the receiver does not know the witness until later in the protocol.

We realize the first one using WE as a RO-based Key Encapsulation Mechanism (KEM). That is, when encrypting a message M, WE only hides an ephemeral key r used to mask M as $\mathsf{H}(r) \oplus M$. In this way the reduction can observe *which* WE ciphertexts an adversary was able to decrypt, thus learning the witness. To program σ instead we encrypt the garbled circuit with the ROM and ephemeral key s. To ensure s is available *only* after a valid m is extracted, we additively secret share s into $s_1, \ldots, s_n$ and encrypt s_i along with the labels $X_{i,0}, X_{i,1}$ for the i-th bit. In this way, once s can be reconstructed, a reduction has enough room to program the RO, so it decrypts to a simulated garbling set to return the desired σ.

We finally remark that due to the heavy dependence on the ROM, this construction *does not* allow proving key correctness (in the ROM). This property is however required by the AS compiler to ensure pre-signature verifiability. For this reason we further discuss an algebraic variant of our scheme in the Bilinear Generic Group Model where key correctness can be shown through e.g. Groth-Sahai proofs [GS08].

1.3 Related Work

Adaptor Signatures. Pioneered by the work of Aumayer et al. [AEE+21] who constructed AS for Schnorr signatures [Sch91] and ECDSA, a long line of works realized AS for various signature schemes. To name a few, Ersig et al. [EEE20] constructed AS for the lattice-based scheme LAS (a simplified version of Delithium [DKL+18]), Tairi et al. [TMM21] presented a AS for the isogeny-based CSI-FiSh [BKV19], and Klamti and Hasan [KH22] put forward AS for the code-based Wave [DST18]. All those constructions exploit the concrete structure the underlying scheme to embed secrets into the signatures. Typically, such embedding is obtained though homomorphic operations on signature's randomness. As a result, only a restricted class of hard relations is supported by these schemes.

The first work looking for more generic AS constructions is that of Erwig et al. [EFH+21]. On the negative side, they proved AS to be impossible for unique signatures. On the positive side, they showed that scheme resulting from identification schemes with some homomorphic properties can support AS. This notably includes Schnorr, Katz-Wang [KW03] and Guillou-Quisquater [GQ90]. Recently, Gerhart et al. [GSST24] improved the result of Erwig et al. They put forward the notion of *dichotomic signatures* that, as they say, cover the essential properties needed to build AS. Their construction applies to several schemes including Camenisch-Lysyanskaya [CL03], Boneh-Boyen-Shacham [BBS04] and Waters signatures signatures [BSW06]. As before, the hard relations supported by these construction are tightly restricted.

More directly related to our work are the results of Dai et al. [DOY22] and Liu et al. [LIZ24]. Both transform *any* EUF-CMA secure signature to an AS for a modified underlying signing scheme. As discussed, they both augment (as we do) the signed message m by some string α, a practically justifiable modification. Their constructions however are not scriptless – as the verification procedure needs to be altered substantially. We further remark that both our construction

and the one of Liu et al. hide the witness while the one in Dai et al. reveals the witness in clear.

Finally we compare the class of relations achieved. Despite the aforementioned limitations, [DOY22, LIZ24] are the only known construction to support any NP relation. Both our results eventually yields AS for UP, i.e. NP relations with unique witness (without any extractability requirement in the second one). Remarkably, such (mild) limitation is actually shared with several other results [TMM21, AEE$^+$21, KH22] and [GSST24, Theorem 3], whose relations are in UP. The only exception is [EEE20] from lattices, although they realize a somewhat weaker notion of AS were witnesses used to adapt have smaller norm than extracted ones. We finally stress that, to the best of our knowledge, UP relations suffices for existing applications.

Functional Signatures. Functional Signatures were originally introduced by Boyle et al. [BGI14]. Two concurrent works also defined very similar notions. Namely, Bellare and Fuchsbauer [BF14] put forward *policy-based* signatures (that is FS for predicates), and Backers et al. [BMS16] defined functional signatures with additional delegation capabilities. The latter notion appears to be the most general, encompassing sanitizable signatures [ACdMT05] identity based signatures [Sha84] blind signatures [Cha83] and more. From a feasibility point of view [BGI14] already proved FS to be realizable from one way function, or from NIZK arguments of knowledge when further requiring function privacy.

Due to our particular application, i.e. realizing *almost scriptless* AS, the mere existence of FS does not suffice to us. Indeed, existing construction require modification to the verification algorithm of the underlying signature scheme. Our construction instead realizes FS generically from any signature, while keeping the verification procedure intact (i.e., we achieve almost scriptless FS). To the best of our knowledge, such problem has not been investigated yet in the literature. We finally remark that our significantly constrained setting is the main reason why we need to resort of powerful tools, such as obfuscation or witness encryption.

2 Preliminaries

Notation. $[n]$ denotes the set $\{1, \ldots, n\}$. $\lambda \in \mathbb{N}$ is the security parameter. A function $f : \mathbb{N} \to \mathbb{R}^+$ is *negligible* if it vanishes faster than the inverse of any polynomial. $\mathsf{negl}(\lambda)$ denotes a generic negligible function. Given $\mathcal{A}$ a probabilistic Turing Machine we denote $y \leftarrow \mathcal{A}(x; r)$ its (deterministic) output on input x and random tape r. $y \leftarrow \mathcal{A}(x)$ is short for $y \leftarrow \mathcal{A}(x; r)$ with r being a uniformly sampled. PPT stands for probabilistic polynomial time. Given a set S we denote by $x \leftarrow S$ the uniformly random sampling of an element x from the set S. Unless stated otherwise, we assume *adversaries* in security definitions to be *stateful*, and procedures in a given scheme (e.g. a digital signature) to be *stateless*. $\mathcal{R} \subseteq \mathcal{X} \times \mathcal{W}$ is an NP-relation if it is decidable in PPT time. We further assume all NP-relations to be efficiently sampleable, i.e. that there exists a procedure to draw a statement-witness from $\mathcal{R}$ with some probability distribution.

2.1 Non-Interactive Zero-Knowledge

A Non-Interactive Zero-Knowledge argument (NIZK) for a relation $\mathcal{R}$ is a tuple of three algorithms $(\mathsf{NIZK.G}, \mathsf{NIZK.P}, \mathsf{NIZK.V})$ allowing a prover to convince a verifier about the validity of a statement without leaking further information. Given $\mathsf{crs} \leftarrow \mathsf{NIZK.G}(1^\lambda)$ and $(x, w) \in \mathcal{R}$ a valid statement, the prover can compute a proof running $\pi \leftarrow \mathsf{NIZK.P}(\mathsf{crs}, x, w)$ which can later be verified by $b \leftarrow \mathsf{NIZK.V}(\mathsf{crs}, x, \pi)$. The proof is accepted if $b = 1$, or rejected otherwise. Below we formally recall the main properties we will require (here $\mathcal{L}_\mathcal{R} := \{x \mid \exists w \colon (x, w) \in \mathcal{R}\}$ denotes the language associated with $\mathcal{R}$).

- **Completeness**: $\forall (x, w) \in \mathcal{R}$

$$\Pr\left[1 \leftarrow \mathsf{NIZK.V}(\mathsf{crs}, x, \pi) \;\middle|\; \mathsf{crs} \leftarrow \mathsf{NIZK.G}(1^\lambda),\ \pi \leftarrow \mathsf{NIZK.P}(\mathsf{crs}, x, w)\right] \;=\; 1.$$

- **Adaptive Soundness**: $\forall$ PPT $\mathcal{A}$ there exists ε negligible such that:

$$\Pr\left[x \notin \mathcal{L}_\mathcal{R},\ 1 \leftarrow \mathsf{NIZK.V}(\mathsf{crs}, x, \pi) \;\middle|\; \begin{matrix}\mathsf{crs} \leftarrow \mathsf{NIZK.G}(1^\lambda) \\ (x, \pi) \leftarrow \mathcal{A}(\mathsf{crs})\end{matrix}\right] \;\leq\; \varepsilon(\lambda).$$

- **Zero-Knowledge**: For every PPT adversary $\mathcal{D}$, there exists a PPT simulator Sim and a negligible ε such that for all $(x, w) \in \mathcal{R}$

$$(\mathsf{crs}_0, \mathsf{td}) \leftarrow \mathsf{Sim}(1^\lambda),\ \pi_0 \leftarrow \mathsf{Sim}(\mathsf{td}, x),\ \mathsf{crs}_1 \leftarrow \mathsf{NIZK.G}(1^\lambda),\ \pi_1 \leftarrow \mathsf{NIZK.P}(\mathsf{crs}_1, x, w)$$

$$\Rightarrow \quad \left|\Pr\left[1 \leftarrow \mathcal{D}(\mathsf{crs}_0, \pi_0)\right] - \Pr\left[1 \leftarrow \mathcal{D}(\mathsf{crs}_1, \pi_1)\right]\right| \leq \varepsilon(\lambda).$$

2.2 Indistinguishability Obfuscation

We briefly recall the definition of Indistinguishability Obfuscator [BGI+01], following the notation of [SW14].

Definition 1 (Indistinguishability Obfuscator). *A uniform* PPT *algorithm* iO *is called an Indistinguishability Obfuscator for a circuit class* $\{\mathcal{C}_\lambda\}$ *if:*

- *For all* $\lambda \in \mathbb{N}$*, for all* $C \in \mathcal{C}_\lambda$*, for all inputs* x*, it holds that*

$$\Pr\left[C'(x) = C(x) : C' \leftarrow \mathsf{iO}(\lambda, C)\right] = 1.$$

- *For any* PPT *adversaries* $\mathcal{S}, \mathcal{D}$*, there exists a negligible* ε *such that, given* $(C_0, C_1, \sigma) \leftarrow \mathcal{S}(\lambda)$*, if* $\Pr\left[\forall x, C_0(x) = C_1(x)\right] > 1 - \varepsilon(\lambda)$*, then it holds that*

$$\left|\Pr\left[\mathcal{D}(\sigma, \mathsf{iO}(\lambda, C_0)) = 1\right] - \Pr\left[\mathcal{D}(\sigma, \mathsf{iO}(\lambda, C_1)) = 1\right]\right| \;\leq\; \varepsilon(\lambda).$$

2.3 Witness Encryption

Witness encryption (WE) [GGSW13] is a generalization of its public-key counterpart. Given a relation $\mathcal{R}$, a WE scheme allows encrypting a message m with respect to a statement x, which can later be decrypted given a witness w for x. Formally, a WE scheme consists of two algorithms $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$.

- $c \leftarrow \mathsf{WE.Enc}(1^\lambda, x, m)$ for any statement x and message m.
- $m \leftarrow \mathsf{WE.Dec}(w, c)$ for any witness w and ciphertext c.

Given a WE scheme for a relation $\mathcal{R}$, with associated language $\mathcal{L}$, we require it to satisfy *correctness* and *soundness*.

Definition 2. *A WE scheme* $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ *for a relation* $\mathcal{R}$ *is perfectly correct if for any message* m *and* $(x, w) \in \mathcal{R}$

$$c \leftarrow \mathsf{WE.Enc}(1^\lambda, x, m) \quad \Rightarrow \quad m \leftarrow \mathsf{WE.Dec}(w, c).$$

Definition 3. *A WE scheme* $(\mathsf{WE.Enc}, \mathsf{WE.Dec})$ *for relation* $\mathcal{R}$ *and associated language* $\mathcal{L}$ *satisfies soundness if, for any* PPT *adversary* $\mathcal{A}$*, there exists* ε *negligible such that for any* $x \notin \mathcal{L}$ *and message pair* $m_0 \neq m_1$

$$\left|\Pr\left[1 \leftarrow \mathcal{A}(\mathsf{WE.Enc}(1^\lambda, x, m_0))\right] - \Pr\left[1 \leftarrow \mathcal{A}(\mathsf{WE.Enc}(1^\lambda, x, m_1))\right]\right| \leq \varepsilon(\lambda).$$

Note the original definition in [GGSW13] addresses WE schemes with message space $\{0,1\}$. Our definition is equivalent to theirs up to encrypting bit-by-bit. In security proofs we will actually make use of the following weaker and easier to manipulate multi-challenge variant. A proof appears in the Appendix, Section C.2.

Lemma 1. *If* WE *is a sound witness encryption scheme, then for any* PPT *adversary* $\mathcal{A}$ *executed in the experiment in Figure 2, there exists a negligible* ε *such that*

$$\left|\Pr\left[\mathsf{Exp}^{\mathsf{mwe}}_{0,\mathsf{WE},\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{mwe}}_{1,\mathsf{WE},\mathcal{A}}(1^\lambda) = 1\right]\right| \;\leq\; \varepsilon(\lambda).$$

2.4 Circuit Garbling

Here we recall notation and definition for circuit garbling, introduced in the seminal work of [Yao82], using (mildly adapted) notation from [BHR12]. A garbling scheme for a function family $\mathcal{F}$ is a tuple $(\mathsf{G.Gr}, \mathsf{G.En}, \mathsf{G.Ev}, \mathsf{G.De})$ such that

- $(\widetilde{C}, e, d) \leftarrow \mathsf{G.Gr}(1^\lambda, C)$ returns a garbling $\widetilde{C}$ of $C \in \mathcal{F}$, along with encoding and decoding information e, d.
- $X \leftarrow \mathsf{G.En}(e, x)$ returns an encoding of input $x \in \{0,1\}^n$.
- $Y \leftarrow \mathsf{G.Ev}(\widetilde{C}, X)$ evaluates the garbled $\widetilde{C}$ on encoded input X returning encoded output Y.

$\mathsf{Exp}^{\mathsf{mwe}}_{b,\mathsf{WE},\mathcal{A}}(1^\lambda)$:

1 : $\mathcal{A}(1^\lambda) \to (x_i, m_{i,0}, m_{i,1})_{i=1}^n$
2 : **For** $i \in \{1, \ldots, n\}$:
3 : **If** $x_i \in \mathcal{L}$: $c_i \leftarrow \mathsf{WE.Enc}(1^\lambda, x_i, m_{i,0})$
4 : **Else**: $c_i \leftarrow \mathsf{WE.Enc}(1^\lambda, x_i, m_{i,b})$
5 : $\mathcal{A}(1^\lambda, c_1, \ldots, c_n) \to b'$
6 : **Return** b'

Fig. 2. Multiple Challenge Witness Encryption Game. Note the experiment may run in super-polynomial time to decide $x_i \in \mathcal{L}$.

– $y \leftarrow \mathsf{G.De}(d, Y)$ decodes an encoded output.

The main notions we will consider in this paper are correctness and (non-adaptive, simulation-based) privacy.

Definition 4. *A garbling scheme* G *is perfectly correct if for any* $C \in \mathcal{F}$ *any* $(\widetilde{C}, e, d)$ *in the support of* $\mathsf{G.Gr}(1^\lambda, C)$ *and any input* x *it holds that, calling* $X \leftarrow \mathsf{G.En}(e, x)$ *and* $Y \leftarrow \mathsf{G.Ev}(\widetilde{C}, X)$*, then* $C(x) \leftarrow \mathsf{G.De}(d, Y)$*.*

Definition 5. *A garbling scheme* G *satisfies (non adaptive, simulation based) privacy if there exists a* PPT *simulator* $\mathcal{S}$ *and leakage function* Φ *such that for any* PPT *adversary* $\mathcal{A}$ *executed in the game in Figure 3*

$$\mathsf{Adv}_{\mathcal{A}}(\lambda) \;:=\; \left|\Pr\left[\mathsf{Exp}^{\mathsf{priv}}_{0,\mathsf{G},\mathcal{A}}(1^\lambda) = 1\right] - \Pr\left[\mathsf{Exp}^{\mathsf{priv}}_{1,\mathsf{G},\mathcal{A}}(1^\lambda) = 1\right]\right| \;\leq\; \mathsf{negl}(\lambda).$$

$\mathsf{Exp}^{\mathsf{priv}}_{b,\mathsf{G},\mathcal{A}}(1^\lambda)$:

1 : $\mathcal{A}(1^\lambda) \to (C, x)$ // $C \in \mathcal{F}$
2 : $y \leftarrow C(x)$
3 : $(\widetilde{C}_0, e_0, d_0) \leftarrow \mathsf{G.Gr}(1^\lambda, C)$; $X_0 \leftarrow \mathsf{G.En}(e_0, x)$
4 : $(\widetilde{C}_1, X_1, d_1) \leftarrow \mathcal{S}(1^\lambda, y, \Phi(C))$
5 : **Return** $\mathcal{A}(\widetilde{C}_b, X_b, d_b)$

Fig. 3. Privacy game for garbling G adversary $\mathcal{A}$ challenge bit b and leakage Φ.

Finally, in this work we only consider *projective* garbling schemes, where, garbling a circuit C with n input bits, the encoding information e consists of $2n$ tokens $(X_{1,0}, X_{1,1}, \ldots, X_{n,0}, X_{n,1})$ and $\mathsf{G.En}(e, x) = (X_{i,x_i})_{i=1}^n$.

2.5 Adaptor Signatures

An adaptor signature scheme for a relation $\mathcal{R}$ is a tuple of eight algorithms $\mathsf{AS.Gen}, \mathsf{AS.Sign}, \mathsf{AS.Vfy}, \mathsf{AS.Setup}, \mathsf{AS.pSign}, \mathsf{AS.pVfy}, \mathsf{AS.Adapt}, \mathsf{AS.Ext}$ along with

an ensembles of message spaces $\mathcal{M}_\lambda$, where the first three define a digital signature [GMR88] (which we call the *base signature* of AS) and the remaining are such that: (here $\mathcal{L}_\mathcal{R}$ is the language associated with $\mathcal{R}$)

- $\mathsf{crs} \leftarrow \mathsf{AS.Setup}(1^\lambda)$ generates a common reference string crs.
- $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{crs}, \mathsf{sk}, m, x)$ returns the pre-signature $\widetilde{\sigma}$ on message $m \in \mathcal{M}_\lambda$ and statement $x \in \mathcal{L}_\mathcal{R}$.
- $b \leftarrow \mathsf{AS.pVfy}(\mathsf{crs}, \mathsf{vk}, m, x, \widetilde{\sigma})$ checks the pre-signature validity w.r.t. a common reference string crs, verification key vk, message m and statement x.
- $\sigma \leftarrow \mathsf{AS.Adapt}(\mathsf{vk}, m, \widetilde{\sigma}, w)$ computes a signature σ for $m \in \mathcal{M}_\lambda$ from $\widetilde{\sigma}$ and a witness w.
- $w/\perp \leftarrow \mathsf{AS.Ext}(\mathsf{vk}, m, x, \widetilde{\sigma}, \sigma)$ extracts a witness from a signature σ and pre-signature $\widetilde{\sigma}$.

Note that in contrast to prior works on AS, our syntax includes an additional $\mathsf{AS.Setup}$ algorithm that generates a common reference string crs. This parameter is taken as input by the algorithms related to the alternative signing process of AS, namely by the $\mathsf{AS.pSign}$ and $\mathsf{AS.pVfy}$ algorithms, but is not taken as input by any other of the AS algorithms. In particular, $(\mathsf{AS.Gen}, \mathsf{AS.Sign}, \mathsf{AS.Vfy})$ forming the base signature of the AS scheme is independent of the crs. Looking ahead, this will be important when talking about scriptlessness of an AS scheme formally as we do in Section 3.

Along with the correctness of the base signature scheme, AS are required to satisfy *pre-signature correctness*.

Definition 6. *Given an AS scheme, for any* $(\mathsf{vk}, \mathsf{sk})$ *in the range of* $\mathsf{AS.Gen}(1^\lambda)$*, message* $m \in \mathcal{M}_\lambda$*, pair* $(x, w) \in \mathcal{R}$*, and* crs *in the range of* $\mathsf{AS.Setup}$*, computing pre-signature* $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{crs}, \mathsf{sk}, m, x)$*, signature* $\sigma \leftarrow \mathsf{AS.Adapt}(\mathsf{vk}, m, \widetilde{\sigma}, w)$ *and* $w' \leftarrow \mathsf{AS.Ext}(\mathsf{vk}, m, x, \widetilde{\sigma}, \sigma)$ *then*

$$1 \leftarrow \mathsf{AS.pVfy}(\mathsf{crs}, \mathsf{vk}, m, x, \widetilde{\sigma}), \qquad 1 \leftarrow \mathsf{AS.Vfy}(\mathsf{vk}, m, \sigma), \qquad (x, w') \in \mathcal{R}.$$

Regarding security, several papers over the last years focused on refining and simplifying the main properties required for applications. In this work we consider *pre-signature adaptability* [AEE+21], *extractability* [DOY22], and *unlinkability* [DOY22]. Other properties proposed in the literature, such as unforgeability and extractability in [AEE+21], or witness hiding in [LIZ24], were shown to be implied by the three above. We finally do not explicitly consider *pre-signature soundness* [GSST24], requiring that pre-signatures for false statement are always rejected. As stated in [GSST24] this is unachievable if testing membership in $\mathcal{L}_\mathcal{R}$ is hard. Conversely if $\mathcal{L}_\mathcal{R}$ can be efficiently decided, pre-signature soundness is obtained up further checking in $\mathsf{AS.pVfy}$ whether $x \in \mathcal{L}_\mathcal{R}$.

Definition 7. *An adaptor signature* AS *with respect to* $\mathcal{R}$ *has pre-signatures adaptability if for any* vk*,* crs *in the range of* $\mathsf{AS.Setup}$*,* $(x, w) \in \mathcal{R}$*, message* m *and pre-signature* $\widetilde{\sigma}$

$$\mathsf{AS.pVfy}(\mathsf{crs}, \mathsf{vk}, m, x, \widetilde{\sigma}) = 1 \quad \Rightarrow \quad \mathsf{AS.Vfy}\left(\mathsf{vk}, m, \mathsf{AS.Adapt}(\mathsf{vk}, m, \widetilde{\sigma}, w)\right) = 1.$$

AS *has computational pre-signature adaptability if, for any* $m \in \mathcal{M}_\lambda$, $(x, w) \in \mathcal{R}$, crs *in the range of* $\mathsf{AS.Setup}$ *and* PPT *adversary* $\mathcal{A}$ *such that* $(\mathsf{vk}, \widetilde{\sigma}) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs}, m, x, w)$*:*

$$\Pr\left[\mathsf{AS.pVfy}(\mathsf{crs}, \mathsf{vk}, m, x, \widetilde{\sigma}) = 1,\ \mathsf{AS.Vfy}\left(\mathsf{vk}, m, \mathsf{AS.Adapt}(\mathsf{vk}, m, \widetilde{\sigma}, w)\right) = 0\right] \leq \mathsf{negl}(\lambda).$$

Definition 8. *An adaptor signature* AS *with respect to a relation* $\mathcal{R}$ *is witness extractable if for any* PPT *adversary* $\mathcal{A}$ *there exists* ε *negligible such that*

$$\mathsf{Adv}_{\mathcal{A}}(\lambda) \coloneqq \Pr\left[\mathsf{Exp}^{\mathsf{we}}_{\mathsf{AS},\mathcal{R},\mathcal{A}}(1^\lambda) = 1\right] \leq \varepsilon(\lambda)$$

where the witness extractability game is defined in Figure 4.

$\mathsf{Exp}^{\mathsf{we}}_{\mathsf{AS},\mathcal{R},\mathcal{A}}(1^\lambda)$:

1 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{AS.Gen}(1^\lambda)$ and $\mathsf{crs} \leftarrow \mathsf{AS.Setup}(1^\lambda)$
2 : $b \leftarrow 1$ and initialize empty sets $\mathcal{S}_{\mathsf{AS}}$, $\mathcal{T}_{\mathsf{AS}}[m]$
3 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2}(\mathsf{crs}, \mathsf{vk})$
4 : **for all** $(x, \widetilde{\sigma}) \in \mathcal{T}_{\mathsf{AS}}[m^*]$:
5 : $\quad w \leftarrow \mathsf{AS.Ext}(\mathsf{vk}, m^*, x, \widetilde{\sigma}, \sigma^*)$
6 : $\quad$ if $(x, w) \in \mathcal{R}$ set $b \leftarrow 0$
7 : **Return** 1 if the following conditions hold:
8 : $\quad 1 \leftarrow \mathsf{AS.Vfy}(\mathsf{vk}, m^*, \sigma^*) \ \wedge\ m^* \notin \mathcal{S}_{\mathsf{AS}} \ \wedge\ b = 1$

$\mathcal{O}_1(m)$: // Signature Oracle

1 : $\sigma \leftarrow \mathsf{AS.Sign}(\mathsf{sk}, m)$
2 : $\mathcal{S}_{\mathsf{AS}} \leftarrow \mathcal{S}_{\mathsf{AS}} \cup \{m\}$
3 : **Return** σ

$\mathcal{O}_2(m, x)$: // Pre-signature Oracle

1 : $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{crs}, \mathsf{sk}, m, x)$
2 : $\mathcal{T}_{\mathsf{AS}}[m] \leftarrow \mathcal{T}_{\mathsf{AS}}[m] \cup \{(x, \widetilde{\sigma})\}$
3 : **Return** $\widetilde{\sigma}$

Fig. 4. Witness Extractability game, adapted from [DOY22]. Note that $b = 1$ holds if none of the presignatures on m^* can be used to extract a witness from σ^*.

Definition 9. *An adaptor signature* AS *with respect to a relation* $\mathcal{R}$ *is unlinkable if for any* PPT *adversary* $\mathcal{A}$ *there exists a negligible* ε *such that*

$$\mathsf{Adv}_{\mathcal{A}}(\lambda) \coloneqq \left|\Pr\left[\mathsf{Exp}^{\mathsf{ul}}_{\mathsf{AS},\mathcal{R},\mathcal{A}}(1^\lambda) = 1\right] - 1/2\right| \leq \varepsilon(\lambda).$$

2.6 Functional Signatures

Here we recall the notion of functional signature (for the special case of predicates) introduced in [BGI14], which we extend to predicates relative to a public

$\mathsf{Exp}^{\mathsf{ul}}_{\mathsf{AS},\mathcal{R},\mathcal{A}}(1^\lambda)$:

1 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{AS.Gen}(1^\lambda)$
2 : $\mathsf{crs} \leftarrow \mathsf{AS.Setup}(1^\lambda)$
3 : $b \leftarrow \{0,1\}$
4 : $b' \leftarrow \mathcal{A}^{\mathsf{AS.Sign},\ \mathsf{AS.pSign},\ \mathcal{O}}(\mathsf{crs}, \mathsf{vk})$
5 : **Return** $b = b'$

$\mathcal{O}(m, x, w)$:

1 : // Challenge Oracle, called only once
2 : **If** $(x, w) \notin \mathcal{R}$: **Return** $\perp$
3 : $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{crs}, \mathsf{sk}, m, x)$
4 : $\sigma_0 \leftarrow \mathsf{AS.Sign}(\mathsf{sk}, m)$
5 : $\sigma_1 \leftarrow \mathsf{AS.Adapt}(\mathsf{vk}, m, \widetilde{\sigma}, w)$
6 : **Return** σ_b

Fig. 5. Unlinkability game from [DOY22].

CRS. Informally, from a master secret key sk it is possible to derive *functional* keys sk_p for a given predicate p which allow signing only those messages satisfying p. More precisely, a functional signature scheme consists of six procedures $\mathsf{FS.Setup}, \mathsf{FS.mskSetup}, \mathsf{FS.KeyGen}, \mathsf{FS.mskSign}, \mathsf{FS.Sign}, \mathsf{FS.Vfy}$ along with an ensemble of message spaces $\mathcal{M}_\lambda$ such that, for a given predicate family $\mathcal{P}$

- $\mathsf{crs} \leftarrow \mathsf{FS.Setup}(1^\lambda)$ samples a common reference string;
- $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{FS.mskSetup}(1^\lambda)$ generates verification and master secret keys;
- $\mathsf{sk}_p \leftarrow \mathsf{FS.KeyGen}(\mathsf{crs}, \mathsf{sk}, p)$ returns sk_p for the predicate $p \in \mathcal{P}$;
- $\sigma \leftarrow \mathsf{FS.mskSign}(\mathsf{sk}, m)$ signs the message m with the master secret key sk.
- $\sigma \leftarrow \mathsf{FS.Sign}(\mathsf{sk}_p, m)$ signs the message m with a functional key sk_p.
- $b \leftarrow \mathsf{FS.Vfy}(\mathsf{vk}, m, \sigma)$ verifies the message/signature pair.

We say that the three algorithms $(\mathsf{FS.mskSetup}, \mathsf{FS.mskSign}, \mathsf{FS.Vfy})$ form the *base signature* of the FS scheme. A functional signature scheme FS is said to be correct if its base signature is correct and the following holds: for any crs and $(\mathsf{vk}, \mathsf{sk})$ in the range of $\mathsf{FS.Setup}(1^\lambda)$ and $\mathsf{FS.mskSetup}(1^\lambda)$ respectively, predicate p and message m so that $p(\mathsf{crs}, m) = 1$, key $\mathsf{sk}_p \leftarrow \mathsf{FS.KeyGen}(\mathsf{crs}, \mathsf{sk}, p)$ and $\sigma \leftarrow \mathsf{FS.Sign}(\mathsf{sk}_p, m)$ then $\mathsf{FS.Vfy}(\mathsf{vk}, m, \sigma) = 1$.

The main security notion for functional signatures is *unforgeability*, which extends the standard notion.

Definition 10. *A functional signature scheme* FS *is unforgeable if for any* PPT *adversary* $\mathcal{A}$ *executed in the game in Figure 6 there exists a negligible* ε *such that*

$$\mathsf{Adv}_{\mathcal{A}}(\lambda) \ := \ \Pr\left[\mathsf{Exp}^{\mathsf{uf}}_{\mathsf{FS},\mathcal{A}}(1^\lambda) = 1\right] \ \leq \ \varepsilon(\lambda).$$

Another notion usually defined for functional signature is that of *function privacy*. This requires that signatures of the same message obtained either from $\mathsf{FS.mskSign}$ or $\mathsf{FS.Sign}$ cannot be distinguished even after the master key is revealed. In this work we consider a weaker notion which we call *one-time function privacy*. Informally, security holds only if the key is used only once.

$\mathsf{Exp}^{\mathsf{uf}}_{\mathsf{FS},\mathcal{A}}(1^\lambda)$:

1 : $\mathsf{crs} \leftarrow \mathsf{FS.Setup}(1^\lambda)$, $(\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{FS.mskSetup}(1^\lambda)$ and $\mathcal{T}_{\mathsf{FS}}$, $\mathcal{S}_{\mathsf{FS}} \coloneqq \varnothing$
2 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_1,\mathcal{O}_2}(\mathsf{crs},\mathsf{vk})$
3 : **Return** 1 if the following conditions are satisfied:
4 : $\quad \mathsf{FS.Vfy}(\mathsf{vk}, m^*, \sigma^*) = 1$
5 : $\quad m^* \notin \mathcal{S}_{\mathsf{FS}}$
6 : $\quad \bigwedge_{p \in \mathcal{T}_{\mathsf{FS}}} p(m^*) = 0$

$\mathcal{O}_1(p)$: // Functional Keys Oracle

1 : $\mathsf{sk}_p \leftarrow \mathsf{FS.KeyGen}(\mathsf{crs},\mathsf{sk},p)$
2 : $\mathcal{T}_{\mathsf{FS}} \leftarrow \mathcal{T}_{\mathsf{FS}} \cup \{p\}$
3 : **Return** sk_p

$\mathcal{O}_2(p,m)$: // Signing Oracle

1 : **If** $p = \top$:
2 : $\quad \sigma \leftarrow \mathsf{FS.mskSign}(\mathsf{sk}, m)$
3 : **Else**:
4 : $\quad \mathsf{sk}_p \leftarrow \mathsf{FS.KeyGen}(\mathsf{crs},\mathsf{sk},p)$
5 : $\quad \sigma \leftarrow \mathsf{FS.Sign}(\mathsf{sk}_p, m)$
6 : $\mathcal{S}_{\mathsf{FS}} \leftarrow \mathcal{S}_{\mathsf{FS}} \cup \{m\}$
7 : **Return** σ

Fig. 6. Functional Signature Unforgeability.

Definition 11. *A functional signature scheme* FS *has one-time function privacy if for any* PPT *adversary* $\mathcal{A}$ *executed in the game in Figure 7 there exists a negligible* ε *such that*

$$\mathsf{Adv}_{\mathcal{A}}(\lambda) \;\coloneqq\; \left|\Pr\left[\mathsf{Exp}^{\mathsf{fp}}_{\mathsf{FS},\mathcal{A}}(1^\lambda) = 1\right] - 1/2\right| \;\leq\; \varepsilon(\lambda).$$

Remark 1. To keep the exposition cleaner, we assume the crs to be implicitly provided as input to predicates p associated with functional keys sk_p. I.e. sk_p is sufficient to sign m if $p(m) = 1$.

3 Scriptless Property for Signatures

As discussed in the introduction, an important property of adaptor signatures for practical applications is their scriptlessness with respect to some "standardized" signature scheme. Intuitively, this means that the verification algorithm of the AS base signature is the same as the verification algorithm of the respective "standardized" signature scheme. While the term *scriptless* frequently appears in the adaptor signature literature (as well as in the blockchain community blogs and discussions) it has never been defined formally.

$\mathsf{Exp}^{\mathsf{fp}}_{\mathsf{FS},\mathcal{A}}(1^\lambda)$:

1 : $\mathsf{crs} \leftarrow \mathsf{FS.Setup}(1^\lambda)$, $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{FS.mskSetup}(1^\lambda)$ and $b \leftarrow \{0,1\}$
2 : $(m, p) \leftarrow \mathcal{A}(\mathsf{vk}, \mathsf{sk}, \mathsf{crs})$
3 : **If** $p(m) = 0$: **Return** 0
4 : $\mathsf{sk}_p \leftarrow \mathsf{FS.KeyGen}(\mathsf{crs}, \mathsf{sk}, p)$
5 : $\sigma_0 \leftarrow \mathsf{FS.mskSign}(\mathsf{sk}, m)$, $\sigma_1 \leftarrow \mathsf{FS.Sign}(\mathsf{sk}_p, m)$
6 : $b' \leftarrow \mathcal{A}(\mathsf{vk}, \mathsf{sk}, \sigma_b)$ // $\mathcal{A}$ does not receive sk_p
7 : **Return** $b = b'$

Fig. 7. One-Time Function Privacy for functional signature schemes.

3.1 Definition

The contribution of this section is to fill the gap and put forward a formal definition of scriptlessness. We choose to define this notion more generally than how it was used in the adaptor signature literature until now. Firstly, we define scriptlessness for any signature scheme (not only for adaptor signatures). Secondly, our definition allows to quantify "how scriptless" a signature scheme S is with respect to another signature scheme S'. We capture this latter requirement by asking the two verification procedures to be the same, up to the composition with a *translator* circuit $\mathcal{T}$, mapping (m, σ) for S to (m', σ') for S'. The more expressive we allow $\mathcal{T}$ to be, the weaker the resulting notion we obtain.

Definition 12 (Scriptlessness). *We say that a signature scheme* $\mathsf{S} = (\mathsf{S.Gen}, \mathsf{S.Sign}, \mathsf{S.Vfy})$ *is* $\mathcal{C}$-scriptless *w.r.t. a signature scheme* $\mathsf{S}' = (\mathsf{S}'.\mathsf{Gen}, \mathsf{S}'.\mathsf{Sign}, \mathsf{S}'.\mathsf{Vfy})$ *and a circuit class* $\mathcal{C}$ *if, calling* n, n' *and* s, s' *the message and signature bit length of* S, S' *respectively, then:*

1. $(\mathsf{vk}_0, \mathsf{sk}_0) \leftarrow \mathsf{S.Gen}(1^\lambda)$ *and* $(\mathsf{vk}_1, \mathsf{sk}_1) \leftarrow \mathsf{S}'.\mathsf{Gen}(1^\lambda)$ *implies* $\mathsf{vk}_0, \mathsf{vk}_1$ *are identically distributed.*
2. *There exists* $\mathcal{T} \in \mathcal{C}$ *s.t. for every pair of strings* $(m, \sigma) \in \{0,1\}^{n \times s}$ *and verification key* vk *it holds that*

$$\mathsf{S.Vfy}(\mathsf{vk}, m, \sigma) = \mathsf{S}'.\mathsf{Vfy}(\mathsf{vk}, \mathsf{Proj}_1(\mathcal{T}(m, \sigma)), \mathsf{Proj}_2(\mathcal{T}(m, \sigma))),$$

where $\mathsf{Proj}_1 : \{0,1\}^{n'+s'} \to \{0,1\}^{n'}$ *and* $\mathsf{Proj}_2 : \{0,1\}^{n'+s'} \to \{0,1\}^{s'}$ *are the projections respectively on the first* n' *and last* s' *bits,*

For the sake of simplicity we will omit when clear from context n, s, n', s'. Moreover, we will avoid explicitly referencing the projection and instead write with abuse of notation

$$\mathsf{S}'.\mathsf{Vfy}(\mathsf{vk}, \mathcal{T}(m, \sigma)) \;=\; \mathsf{S}'.\mathsf{Vfy}(\mathsf{vk}, \mathsf{Proj}_1(\mathcal{T}(m, \sigma)), \mathsf{Proj}_2(\mathcal{T}(m, \sigma))).$$

Equipped with the general definition of scriptlessness, we can now naturally lift the notion to adaptor signatures.

Definition 13 (Scriptlessness for AS). *We say that an adaptor signature scheme* AS *is* $\mathcal{C}$-scriptless *w.r.t. a signature scheme* S' *if the base signature of* AS*, i.e.,* $(\mathsf{AS.Gen}, \mathsf{AS.Sign}, \mathsf{AS.Vfy})$*, is* $\mathcal{C}$-scriptless *w.r.t.* S'*.*

In a similar fashion, we can define scriptlessness for functional signatures.

Definition 14 (Scriptlessness for FS). *We say that a functional signature scheme* FS *is* $\mathcal{C}$-scriptless *w.r.t. a signature scheme* S' *if the base signature of* FS*, i.e.,* $(\mathsf{FS.mskSetup}, \mathsf{FS.mskSign}, \mathsf{FS.Vfy})$*, is* $\mathcal{C}$-scriptless *w.r.t.* S'*.*

Remark 2. It is useful to realize, that our goal is to show that for any external signature scheme Σ, there exists an AS scheme AS that is almost scriptless w.r.t. Σ. The order of the quantifiers is important here. Note that any AS is by definition fully scriptless w.r.t. its own base signature. The same is true for FS.

3.2 Degrees of Scriptlessness

Let us demonstrate that the scriptlessness definition is fine-grained enough to capture the relevant degrees of scriptlessness that appear in the literature.

Full scriptlessness. If the circuit family $\mathcal{C}$ *only* contains the identity circuit, we obtain the strongest scriptlessness notion possible, as it implies $\mathsf{S.Vfy} = \mathsf{S}'.\mathsf{Vfy}$. This is the implicit notion that appeared under the term *scriptless* in the literature so far and also the notion used in the impossibility result of Erwig et al. [EFH+21].

Almost scritplessness. Throughout this work, we will be interested in $(\mathcal{C}_{\mathsf{loc\text{-}1}})$-scriptless, were $\mathcal{C}_{\mathsf{loc\text{-}1}}$ is the set of all circuits of locality-1. This formally means that each output bit only depends on at most one input bit (predetermined by the circuit topology). In this work, we will only need circuits $\mathcal{T}$ performing the following simple transformation

$$\mathsf{Proj}_1(\mathcal{T}\,(m,\,\widehat{\sigma}||r)) = m||r, \qquad \mathsf{Proj}_2(\mathcal{T}\,(m,\,\widehat{\sigma}||r)) = \widehat{\sigma}$$

It is immediate to see that $\mathcal{T} \in \mathcal{C}_{\mathsf{loc\text{-}1}}$. To keep notation lighter, any future reference to *almost scriptless* property should be interpreted as $(\mathcal{C}_{\mathsf{loc\text{-}1}})$-scriptless.

Weaker notions. Our scriptlessness definition is general enough to capture the generic constructions in [DOY22, LIZ24] that for any NP relation $\mathcal{R}$, turns any signature scheme S' into an adaptor signature AS for $\mathcal{R}$.

Let us focus on the simpler construction [DOY22] (the argument for [LIZ24] is analogous). An adapted signature for a message m with respect to a statement witness pair (x, w) is of the form $\sigma = (\widehat{\sigma}, x, w)$. The algorithm $\mathsf{AS.Vfy}$ checks whether $\mathsf{S}'.\mathsf{Vfy}(\mathsf{vk}, m\|x, \widehat{\sigma}) = 1$ and $(x, w) \in \mathcal{R}$. Specifically the extra check $(x, w) \in \mathcal{R}$ could be emulated by the translation circuit setting

$$\mathcal{T}(m, (\widehat{\sigma}, x, w)) \;=\; \begin{cases} (m\|x, \widehat{\sigma}) & \text{If } (x, w) \in \mathcal{R} \\ \bot & \text{Otherwise.} \end{cases}$$

However, if $\mathcal{R}$ is a hard relation, testing membership in $\mathcal{R}$ requires significantly more expressive circuits than those with input locality 1. Hence, the constructions in [DOY22, LIZ24] would not be fully or even almost scriptless w.r.t. the signature scheme S' they are built from.

4 Adaptor Signatures From Functional Signatures

In this section we present our generic construction of adaptor signature from functional signature schemes which in turn can be constructed from any EUF-CMA secure signature schemes as we show in later sections. As a result, we obtain almost scriptless AS for any EUF-CMA secure signature scheme.

The way known barriers for unique or randomizable/sanitizable signatures are bypassed is by signing *extended messages*. Specifically, when asked to sign a message m, we require that a random string r is sampled and that a signature $\widehat{\sigma}$ for $M = (m, r)$ is computed. In order to formally capture such message extension in the AS framework, we actually model the scheme to associate $\sigma = (\widehat{\sigma}, r)$ as a signature for m. Note, however, that there is an easy way to express such message signature pairs as valid message signature pairs of the underlying signature scheme simply by removing the random string from the signature and attaching it to the message. Hence, our scheme is almost scriptless w.r.t. the underlying signature scheme.

To turn this scheme into an AS, we assume the starting signature to be functional. In this case a pre-signature $\widetilde{\sigma}$ for a message m and statement x will be a functional key sk_p, where $p(m', r')$ is satisfied when $m' = m$ and $(x, r') \in \mathcal{R}$. Intuitively, witness extractability follows from the unforgeability of the functional signature scheme. Indeed any signature (m^*, r^*) the adversary can compute, must be such that $p(m^*, r^*) = 1$ for some queried predicate p. In particular $m^* = m$ and $(x, r^*) \in \mathcal{R}$ for a message/statement pair previously requested to the pre-signature oracle, meaning that r^* is a witness for x. Finally, to achieve pre-signature adaptability by adding a NIZK certifying sk_p was computed correctly. Formally, a NIZK for the following relation is used:

$$\mathcal{R}_{\mathsf{key}} \;=\; \left\{ ((\mathsf{vk}, p, \mathsf{sk}_p), (\mathsf{sk}, R)) \;:\; \begin{array}{l} (\mathsf{vk}, \mathsf{sk}) = \mathsf{FS.mskSetup}(1^\lambda; \mathsf{sk}), \\ \mathsf{sk}_p = \mathsf{FS.KeyGen}(\mathsf{sk}, p; R) \end{array} \right\} \tag{1}$$

where we assume without loss of generality that sk consists of the random coins used to generate vk.

Note the above construction would trivially leak the witness. We avoid this through a random shifting. Formally, for any NP-relation $\mathcal{R} \subseteq \{0,1\}^{\ell'} \times \{0,1\}^{\ell}$, we define $p_{m,x,\rho}(m', r') = (m{=}m' \wedge (x, r \oplus \rho) \in \mathcal{R})$ for any message m, statement $x \in \{0,1\}^{\ell'}$ and random shift $\rho \in \{0,1\}^{\ell}$, and let

$$\mathcal{P}_{\mathcal{R}} \;=\; \{p_{m,x,\rho} \;:\; m \in \mathcal{M}_\lambda,\; x \in \{0,1\}^{\ell'},\; \rho \in \{0,1\}^{\ell}\}. \tag{2}$$

In this way, if ρ is further added in the pre-signature, only $r' = w \oplus \rho$ is leaked with a signature and, most importantly, this becomes unlinkable since the ran-

dom string added to the signed message has the same distribution observed in non-adapted signatures. A full description is provided in Figure 8.

$\mathsf{AS.Setup}(1^\lambda)$:

1 : $\mathsf{crs}_1 \leftarrow \mathsf{NIZK.G}(1^\lambda)$
2 : $\mathsf{crs}_2 \leftarrow \mathsf{FS.Setup}(1^\lambda)$
3 : **Return** $\mathsf{crs} = (\mathsf{crs}_1, \mathsf{crs}_2)$

$\mathsf{AS.Gen}(1^\lambda)$:

1 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{FS.mskSetup}(1^\lambda)$
2 : **Return** $(\mathsf{vk}, \mathsf{sk})$

$\mathsf{AS.Sign}(\mathsf{sk}, m)$:

1 : $r \leftarrow \{0,1\}^\ell$
2 : $\sigma \leftarrow \mathsf{FS.mskSign}(\mathsf{sk}, m \,||\, r)$
3 : **Return** (σ, r)

$\mathsf{AS.Vfy}(\mathsf{vk}, m, \sigma)$:

1 : Parse $\sigma = (\widehat{\sigma}, r)$:
2 : **Return** $\mathsf{FS.Vfy}(\mathsf{vk}, m \,||\, r, \widehat{\sigma})$

$\mathsf{AS.pSign}(\mathsf{crs}, \mathsf{sk}, m, x)$:

1 : $\rho \leftarrow \{0,1\}^\ell$ and R random tape
2 : $p \coloneqq p_{m,x,\rho}$ and parse $\mathsf{crs} = (\mathsf{crs}_1, \mathsf{crs}_2)$
3 : $\mathsf{sk}_p \leftarrow \mathsf{FS.KeyGen}(\mathsf{crs}_2, \mathsf{sk}, p; R)$
4 : $\pi \leftarrow \mathsf{NIZK.P}(\mathsf{crs}_1, (\mathsf{vk}, p, \mathsf{sk}_p), (\mathsf{sk}, R))$
5 : **Return** $(\mathsf{sk}_p, \pi, \rho)$

$\mathsf{AS.pVfy}(\mathsf{crs}, \mathsf{vk}, m, x, \widetilde{\sigma})$:

1 : Parse $\widetilde{\sigma} = (\mathsf{sk}_p, \pi, \rho)$
2 : Parse $\mathsf{crs} = (\mathsf{crs}_1, \mathsf{crs}_2)$
3 : $p \coloneqq p_{m,x,\rho}$
4 : $b \leftarrow \mathsf{NIZK.V}(\mathsf{crs}_1, (\mathsf{vk}, p, \mathsf{sk}_p), \pi)$
5 : **Return** b

$\mathsf{AS.Adapt}(\mathsf{vk}, m, \widetilde{\sigma}, w)$:

1 : Parse $\widetilde{\sigma} = (\mathsf{sk}_p, \pi, \rho)$
2 : $r \leftarrow w \oplus \rho$
3 : $\widehat{\sigma} \leftarrow \mathsf{FS.Sign}(\mathsf{sk}_p, m \,||\, r)$
4 : **Return** $\sigma = (\widehat{\sigma}, r)$

$\mathsf{AS.Ext}(\mathsf{vk}, m, x, \widetilde{\sigma}, \sigma)$:

1 : Parse $\widetilde{\sigma} = (\mathsf{sk}_p, \pi, \rho)$
2 : Parse $\sigma = (\widehat{\sigma}, r)$
3 : **Return** $r \oplus \rho$

Fig. 8. AS for a relation $\mathcal{R}$ from a Functional Signature scheme for predicates $\mathcal{P}_\mathcal{R}$.

Theorem 1. *For any NP-relation $\mathcal{R}$, if* FS *is a perfectly correct FS scheme for predicates $\mathcal{P}_\mathcal{R}$ and* NIZK *an adaptively secure NIZK for $\mathcal{R}_{\mathsf{key}}$, then the scheme* AS *in Figure 8 is a pre-signature adaptable and extractable adaptor signature. Moreover, if* FS *is one-time function private (Definition 11), then* AS *is unlinkable.*

Due to space limitations, we provide only a proof sketch here and refer the reader to Appx. C.1 for the formal proof.

Witness Extractability. We reduce directly to unforgeability of FS. Given $\mathcal{A}$ breaking witness extractability of our scheme, the reduction $\mathcal{B}$ simulates signatures with its signature oracle (up to sampling the randomness r padded to m)

and pre-signatures by simulating the NIZK, sampling a random ρ and querying sk_p for $p = p_{m,x,\rho}$. When $\mathcal{A}$ gives a forgery $(m^*, \sigma^*) = (m^*, (\widehat{\sigma}^*, r^*))$ then it wins if m^* was never signed and, for all pre-signature $\widetilde{\sigma}$ queried for (m^*, x), $r^* \oplus \rho$ is not a witness for x, or else the extractor succeeds. Hence $p(m^*, r^*) = 0$ for all predicates asked, and in particular $\widehat{\sigma}^*$ is a valid forgery for $\mathcal{B}$.

Pre-Signature Adaptability. Let $\mathcal{A}$ be an adversary which on input (crs, m, x, w) returns $(\mathsf{vk}, \widetilde{\sigma})$ such that $\widetilde{\sigma}$ is a valid pre-signature, but adapting it with w yields an invalid signature. Since $\mathsf{AS.pVfy}$ accepts $\widetilde{\sigma}$, it satisfies $\widetilde{\sigma} = (\mathsf{sk}_p, \pi, \rho)$ with π being an accepting proof. Assume toward contradiction $(\mathsf{vk}, p, \mathsf{sk}_p)$ is a true statement. Then there exists sk, R such that $(\mathsf{vk}, \mathsf{sk})$ is a valid key pair, and $\mathsf{sk}_p = \mathsf{FS.KeyGen}(\mathsf{crs}_2, \mathsf{sk}_p, p; R)$. Perfect correctness of FS implies that the adapted signature $(\widehat{\sigma}, r)$ is accepting, that is a contradiction. Therefore $\mathcal{A}$ can readily be turned into a (non-uniform) adversary for the NIZK adaptive soundness.

Unlinkability. Given $\mathcal{A}$ breaking unlinkability, we define $\mathcal{B}$ breaking one-time function privacy. On input $(\mathsf{vk}, \mathsf{sk})$ $\mathcal{B}$ runs $\mathcal{A}(\mathsf{vk})$ using sk to simulate both signing and pre-signing oracles. Finally, when $\mathcal{A}$ queries (m, x, w) to the challenger, $\mathcal{B}$ queries the predicate $p = p_{m,x,\rho}$ and message $m \,||\, r$ where $r = w \oplus \rho$ for a random ρ. It then gets back $\widehat{\sigma}$ and forwards $\sigma = (\widehat{\sigma}, r)$ to $\mathcal{A}$. In $\mathcal{B}$'s first world with $b = 0$, $\widehat{\sigma}$ is computed as $\mathsf{FS.mskSign}(\mathsf{sk}, m \,||\, r)$ which means that $\sigma = (\widehat{\sigma}, r)$ perfectly follows the distribution of a fresh (i.e. non adapted) signature from $\mathsf{AS.Sign}(\mathsf{sk}, m)$. Conversely when $b = 1$, ρ is uniformly random as $\mathcal{A}$ does not see the pre-signature, and thus so is r. Hence, p corresponds to the predicate that would ve generated in $\mathsf{AS.pSign}$ procedure. In particular $\sigma = (\widehat{\sigma}, r)$ where $\widehat{\sigma} = \mathsf{FS.Sign}(\mathsf{FS.KeyGen}(\mathsf{crs}_2, \mathsf{sk}, p), m \,||\, r)$ perfectly follows the distribution of an adapted pre-signature on $m \,||\, r$. It follows that $\mathsf{Adv}_{\mathcal{A}}(\lambda) = \mathsf{Adv}_{\mathcal{B}}(\lambda) = \mathsf{negl}(\lambda)$.

The following theorem formally states links the scriptlessness of the underlying FS scheme to the scriptlessness of the constructed AS.

Theorem 2. *Given a signature scheme* S*, if* FS *is a* $\mathcal{C}$*-scriptless functional signature for* S*, then* AS *in Fig. 8 is a* $(\mathcal{C} \circ \mathcal{C}_{\mathsf{loc\text{-}1}})$*-scriptless adaptor signature for* S*.*

Proof. Let $\mathsf{S} = (\mathsf{S.Gen}, \mathsf{S.Sign}, \mathsf{S.Vfy})$. First of all, sample $(\mathsf{vk}_0, \mathsf{sk}_0) \leftarrow \mathsf{S.Gen}(1^\lambda)$, $(\mathsf{vk}_1, \mathsf{sk}_1) \leftarrow \mathsf{FS.mskSetup}(1^\lambda)$ and $(\mathsf{vk}_2, \mathsf{sk}_2) \leftarrow \mathsf{AS.Gen}(1^\lambda)$. Then by construction $\mathsf{vk}_1, \mathsf{vk}_2$ are identically distributed, and so are $\mathsf{vk}_0, \mathsf{vk}_1$ due to FS being scriptless for S. Therefore $\mathsf{vk}_0, \mathsf{vk}_2$ follow the same distribution.

Next, let $\mathcal{T} \in \mathcal{C}_{\mathsf{loc\text{-}1}}$ such that $\mathcal{T}(m, \widehat{\sigma}||r) = (m||r, \widehat{\sigma})$. As FS is $\mathcal{C}$-scriptless, there exists $\mathcal{T}^* \in \mathcal{C}$ such that for every m^*, σ^* and vk

$$\mathsf{FS.Vfy}(\mathsf{vk}, m^*, \sigma^*) \;=\; \mathsf{S.Vfy}(\mathsf{vk}, \mathcal{T}^*(m^*, \sigma^*))$$

By construction then for any m, σ and vk

$$\begin{aligned} \mathsf{AS.Vfy}(\mathsf{vk}, m, \sigma) \;&=\; \mathsf{FS.Vfy}(\mathsf{vk}, \mathcal{T}(m, \sigma)) \\ &=\; \mathsf{S.Vfy}(\mathsf{vk}, \mathcal{T}^*(\mathcal{T}(m, \sigma)) \end{aligned}$$

where $\mathcal{T}^* \circ \mathcal{T} \in \mathcal{C} \circ \mathcal{C}_{\mathsf{loc\text{-}1}}$.

Remark 3. Due to $\mathcal{C}_{\mathsf{loc\text{-}1}}$ being closed by composition, if FS is $\mathcal{C}_{\mathsf{loc\text{-}1}}$-scriptless (i.e. *almost scriptless*), so is AS. We note that both our realizations of FS are in fact scriptless (i.e. $\mathcal{C}$ only contains the identity).

5 Almost Scriptless Functional Signatures

5.1 From Obfuscation

In this section a first construction of almost scriptless functional signature is provided. Although no restriction is imposed on the underlying signature, our FS only covers a restricted classes of predicates. Specifically we need to assume that $p \in \mathcal{P}$ admits a unique satisfying input (i.e. the relation $\{(p, w) : p(w) = 1\}$ is in UP). Moreover $\mathcal{P}$ is defined relative to a public crs with backdoor td, so that given td it is possible to efficiently find a witness for any valid $p \in \mathcal{P}$. We formalize this as an *extractable predicates*

Definition 15. *An extractable predicate is a tuple* $(\mathcal{P}, \mathsf{R.Gen}, \mathsf{R.Ext})$ *with* $\mathcal{P}$ *a set of predicates, such that*

- $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{R.Gen}(1^\lambda)$ *samples a common reference string* crs *and trapdoor* td
- $w \leftarrow \mathsf{R.Ext}(\mathsf{td}, p)$ *extracts a supposedly satisfying assignment for* $p(\mathsf{crs}, \cdot)$

An extractable predicate satisfies correctness if there exists a negligible $\varepsilon(\lambda)$ *such that* $\forall p \in \mathcal{P}$*, sampling* $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{R.Gen}(1^\lambda)$*, and extracting* $w \leftarrow \mathsf{R.Ext}(\mathsf{td}, p)$ *it holds that if* $p(\mathsf{crs}, \cdot)$ *is satisfiable, then* $\Pr[p(\mathsf{crs}, w) \neq 1] \leq \varepsilon(\lambda)$.

Examples of non-trivial extractable predicates can be directly derived from any trapdoor permutation, or by attaching a (straight-line extractable) NIZK argument of knowledge, e.g. [Fis05]. We defer the details to the Appendix A.

In the CRS model we can then provide a simple scriptless construction of FS from any signature scheme and a program obfuscator. This is done by setting sk_p as an obfuscation of a circuit returning $\mathsf{S.Sign}(\mathsf{sk}, m)$ only if $p(m) = 1$. The scheme is detailed in Figure 9. Note that although the supported predicate family is required to be extractable, the trapdoor to do so is not available to the signer. In particular, the resulting extractor for AS obtained applying the compiler in Section 4, does not need trapdoor information about the crs (which would otherwise render AS useless).

Theorem 3. *For any secure obfuscator* iO*, signature scheme* $(\mathsf{S.Gen}, \mathsf{S.Sign}, \mathsf{S.Vfy})$ *and extractable predicate* $(\mathcal{P}, \mathsf{R.Gen}, \mathsf{R.Ext})$ *with* $\mathcal{P}$ *admitting unique witness, the FS scheme in Figure 9 is a unforgeable and one-time function private functional signature for the class of predicates* $\mathcal{P}$ *in the CRS model.*

Proof. The proof is divided in two parts, showing unforgeability and one-time function privacy.

$\mathsf{FS.Setup}(1^\lambda)$:

1 : $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{R.Gen}(1^\lambda)$
2 : **Return** crs

$\mathsf{FS.mskSetup}(1^\lambda)$:

1 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{S.Gen}(1^\lambda)$
2 : **Return** $(\mathsf{vk}, \mathsf{sk})$

$\mathsf{FS.KeyGen}(\mathsf{crs}, \mathsf{sk}, p)$:

1 : Sample a random tape r
2 : $\widetilde{C} \leftarrow \mathsf{iO}(C_{\mathsf{sk},\mathsf{crs},p,r})$
3 : **Return** $\mathsf{sk}_p = \widetilde{C}$

$C_{\mathsf{sk},\mathsf{crs},p,r}(m)$:

1 : // obfuscated circuit
2 : **If** $p(\mathsf{crs}, m) = 1$:
3 : **Return** $\mathsf{S.Sign}(\mathsf{sk}, m; r)$
4 : **Else**: **Return** $\perp$

$\mathsf{FS.Sign}(\mathsf{sk}_p, m)$:

1 : Parse $\mathsf{sk}_p = \widetilde{C}$
2 : **Return** $\widetilde{C}(m)$

$\mathsf{FS.Vfy}(\mathsf{vk}, m, \sigma)$:

1 : **Return** $\mathsf{S.Vfy}(\mathsf{vk}, m, \sigma)$

Fig. 9. Almost Scriptless Functional Signature from any signature and Obfuscation, for extractable predicates $(\mathcal{P}, \mathsf{R.Gen}, \mathsf{R.Ext})$ with $\mathcal{P}$ admitting unique witnesses.

Unforgeability. Let $\mathcal{A}$ be an adversary for unforgeability of the scheme in Figure 9. Call q an upper bound on the number of key generation queries it performs. We proceed through a sequence of hybrids $\mathsf{H}_0, \ldots, \mathsf{H}_q$, progressively removing the dependency on sk to compute sk_p leveraging on the crs's trapdoor. Specifically H_i initially set up the CRS as $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{R.Gen}(1^\lambda)$. Then the first i-th queries to $\mathsf{FS.KeyGen}(\mathsf{crs}, \mathsf{sk}, p)$ are replied using $\mathsf{Sim}^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(\mathsf{crs}, \mathsf{td}, p)$ as described in Figure 10.

$\mathsf{Sim}^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(\mathsf{crs}, \mathsf{td}, p)$:

1 : Extract $m \leftarrow \mathsf{R.Ext}(\mathsf{td}, p)$
2 : **If** $p(\mathsf{crs}, m) = 1$:
3 : $\sigma \leftarrow \mathsf{S.Sign}(\mathsf{sk}, m)$
4 : **Return** $\widetilde{C} \leftarrow \mathsf{iO}(C^*_{m,\sigma})$
5 : **Else**:
6 : **Return** $\widetilde{C} \leftarrow \mathsf{iO}(\Omega)$

$C^*_{m,\sigma}(m')$:

1 : **If** $m = m'$:
2 : **Return** σ
3 : **Else**: **Return** $\perp$

Fig. 10. Hybrid key generation procedure. Ω is the constant $\perp$ program, padded to have the same length as C and C^*.

$\mathsf{H}_{i-1} \approx \mathsf{H}_i$. Given a distinguisher $\mathcal{D}$, let $\mathcal{B}$ be an adversary for iO security. Initially $\mathcal{B}$ simulates H_i by sampling $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{R.Gen}(1^\lambda)$ and signature key $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{FS.mskSetup}(\mathsf{crs}, 1^\lambda)$ and running $\mathcal{D}(\mathsf{vk})$. All signing queries, and key queries but the i-th are replies as in $\mathsf{H}_{i-1}, \mathsf{H}_i$.

At the i-th functional key query for p, it prepares two circuits: first, sample a random tape r and set $C_0 = C_{\mathsf{sk},\mathsf{crs},p,r}$. Next, using td it gets $m \leftarrow \mathsf{R.Ext}(\mathsf{td}, p)$. If $p(\mathsf{crs}, m)$ it computes $\sigma \leftarrow \mathsf{S.Sign}(\mathsf{sk}, m; r)$ and sets $C_1 = C^*_{m,\sigma}$. Otherwise it sets $C_1 = \Omega$. Finally, it sends (C_0, C_1) to its challenger, waits for $\widetilde{C}$, and forwards $\mathsf{sk}_p = \widetilde{C}$ to $\mathcal{D}$. Eventually, it returns the same bit $\mathcal{D}$ returned upon halting.

Let Miss be the event in which $p(\mathsf{crs}, \cdot)$ is satisfiable, but an incorrect witness is found. By the correctness of extractable predicates $\Pr[\mathsf{Miss}] \leq \mathsf{negl}(\lambda)$. Conditioning on $\neg\mathsf{Miss}$ instead, either p has no witness, or m is the only input such that $p(\mathsf{crs}, m) = 1$. In the first case, $C_0(m')$ always return $\perp$, meaning that C_0 is functionally equivalent to C_1. In the second case $C_0(m) = \mathsf{S.Sign}(\mathsf{sk}, m; r) = \sigma$ and it return $\perp$ on any other input. Therefore it is again functionally equivalent to C_1. Finally, according to whether $\mathcal{B}$'s challenger obfuscates C_0 or C_1, $\mathcal{B}$ perfectly simulates respectively H_{i-1} or H_i conditioning on $\neg\mathsf{Miss}$. Therefore

$$\mathsf{Adv}_{\mathcal{B}}(\lambda) \geq \mathsf{Adv}_{\mathcal{D}}(\lambda) - \Pr[\mathsf{Miss}] \;\Rightarrow\; \mathsf{Adv}_{\mathcal{D}}(\lambda) \leq \mathsf{negl}(\lambda).$$

Unforgeability holds in H_q. Let $\mathcal{A}$ an adversary for unforgeability in H_q. We reduce it to $\mathcal{B}$ breaking unforgeability for the underlying signature scheme.

Initially $\mathcal{B}(\mathsf{vk})$ samples $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{R.Gen}(1^\lambda)$ and runs $\mathcal{A}(\mathsf{crs}, \mathsf{vk})$. Regular signature queries from $\mathcal{A}$ are simply forwarded to $\mathcal{B}$'s oracle. When $\mathcal{A}$ queries a functional key for predicate p, $\mathcal{B}$ replies with $\mathsf{sk}_p \leftarrow \mathsf{Sim}^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(\mathsf{crs}, \mathsf{td}, p)$ which queries at most a signature of a message m such that $p(\mathsf{crs}, m) = 1$. Finally, when $\mathcal{A}$ returns (m^*, σ^*), so does $\mathcal{B}$.

Assume (m^*, σ^*) is a valid forgery. Then a signature for m^* was never directly requested by $\mathcal{A}$. Moreover, for all predicates $p_1, \ldots, p_q$ requested, $p_i(\mathsf{crs}, m^*) = 0$, meaning that no invocation of Sim requested a signature for m^*. Finally, as the forgery is valid, $\mathsf{FS.Vfy}(\mathsf{vk}, m^*, \sigma^*) = 1$ which is equivalent to $\mathsf{S.Vfy}(\mathsf{vk}, m^*, \sigma^*) = 1$. Therefore we can conclude

$$\mathsf{Adv}_{\mathcal{A}}(\lambda) \;=\; \mathsf{Adv}_{\mathcal{B}}(\lambda) \;\leq\; \mathsf{negl}(\lambda).$$

One-Time Function Privacy. We show this to hold information theoretically. Given $(\mathsf{crs}, \mathsf{vk}, \mathsf{sk})$ correctly sampled and p, m such that $p(\mathsf{crs}, m) = 1$, let $\mathsf{sk}_p \leftarrow \mathsf{FS.KeyGen}(\mathsf{crs}, \mathsf{sk}, p)$, $\sigma_0 \leftarrow \mathsf{FS.Sign}(\mathsf{sk}_p, m)$ and $\sigma_1 \leftarrow \mathsf{S.Sign}(\mathsf{sk}, m)$. By construction $\sigma_0 = \widetilde{C}(m) = \mathsf{S.Sign}(\mathsf{sk}, m; r)$ for a random tape r sampled independently from $\mathsf{crs}, \mathsf{vk}, \mathsf{sk}$. Therefore σ_0, σ_1 follow the same distribution for any fixed $\mathsf{crs}, \mathsf{vk}, \mathsf{sk}$. This in particular implies One-Time function privacy.

Remark 4. Although we describe our construction for predicates admitting only a *unique* witness, we could in fact generalize to $\kappa(\lambda) \in \mathsf{poly}(\lambda)$ many witnesses. Formally this would however require (i) an extraction mechanism $\mathsf{R.Ext}$ to recover *all* possible witnesses; (ii) padding the size of C to match the length of κ signatures.

5.2 From WE and Garbling

In this section we provide another almost scriptless construction of Functional Signature from any signature scheme, relying on simpler assumptions compared to obfuscation. Specifically our construction requires:

1. A witness encryption scheme WE for NP
2. A garbling scheme $(\mathsf{G.Gr}, \mathsf{G.En}, \mathsf{G.De}, \mathsf{G.Ev})$
3. A (programmable) random oracle H

Our high-level approach, for predicates admitting a unique witness, is to adapt non-interactive 2 party computation techniques to let the receiver compute a signature for the only m such that $p(m) = 1$. We face two main challenges. On the one hand, interaction is extremely limited as the signer can only send one message, which is insufficient for e.g. oblivious transfer. On the other hand, the receiver may not know its input m yet. Even worse, such input may not exists.

We solve the above issues using WE, in combination with p unique witness property. Let $X_{1,0}, X_{1,1}, \ldots, X_{n,0}, X_{n,1}$ be the input labels for a garbling of $\mathsf{S.Sign}(\mathsf{sk}, \cdot)$. Given predicate p, witness $m = (m_1, \ldots, m_n) \in \{0,1\}^n$, we wish the receiver to get *only* X_{i,m_i}. This is done encrypting $X_{i,b}$ with WE where resulting ciphertext $c_{i,b}$ can only be opened given a witness m for p such that $m_i = b$. The receiver can then *only* recover an encoding m – while other input labels remain computationally hidden as they were encrypted with a false statement. Formally, the relation we need WE to support is

$$\mathcal{R}_{\mathsf{bit\text{-}sat}} \;=\; \{((p,i,b),w) \;:\; p(w) = 1 \;\wedge\; w_i = b\}.$$

A limitation of the scheme above is that a reduction needs to extract a witness for any p if it exists (e.g. to simulate the garbled circuit). Extractable predicates could be used as in Section 5.1. However, in order to remove such limitation on the supported predicates, we show how the random oracle could also be used to achieve the same result. At a hight level, once sk_p is given, we need two mechanisms in place: one to extract a witness when $\mathcal{A}$ starts computing a signature from sk_p, and another one to dynamically program the garbled circuit *after* the witness is known. Both are achieved as follows:

1. The garbled circuit (along with other parameters) are encrypted with the random oracle and ephemeral symmetric key s. Specifically, given a string x of length $\ell \cdot \lambda$ we denote

$$\mathsf{H.Enc}^{\mathsf{H}}(s,x) \;=\; (\mathsf{H}(s \,||\, 1) \,||\, \ldots \,||\, \mathsf{H}(s \,||\, \ell)) \oplus x, \qquad \mathsf{H.Dec}^{\mathsf{H}} = \mathsf{H.Enc}^{\mathsf{H}}.$$

   In this way, until s is ever queried, a reduction can program the encrypted value.

2. WE is used as a KEM. Specifically, we sample ephemeral keys $r_{i,b}$ set $c_{i,b} = \mathsf{WE.Enc}((p,i,b), r_{i,b})$, and use $\mathsf{H}(r_{i,b})$ as encryption key. In this way, once $r_{i,b}$ is queried to H, the reduction learns a witness w with $w_i = b$ exists.

3. To prevent the adversary from observing the garbled circuit *before* the witness is extracted, we secret-share the key $s = s_1 \oplus \ldots \oplus s_n$ and encrypt them with the ephemeral keys above $\mathsf{H}(r_{i,b}) \oplus s_i$. In this way s is only known *after* $r_{i,b}$ were queried for all $i \in [n]$ and some $b \in \{0,1\}$.

A complete description is provided in Figure 11.

$\mathsf{FS.mskSetup}(1^\lambda)$:

1 : $\mathsf{sk}, \mathsf{vk} \leftarrow \mathsf{S.Gen}(1^\lambda)$
2 : **Return** $(\mathsf{sk}, \mathsf{vk})$

$\mathsf{FS.KeyGen}(\mathsf{sk}, p)$:

1 : Sample a random tape ρ
2 : $(\widetilde{C}, e, d) \leftarrow \mathsf{G.Gr}(\mathsf{S.Sign}(\mathsf{sk}, \cdot\,; \rho))$
3 : Parse $e = (X_{1,0}, \ldots, X_{n,1})$
4 : **For** $i \in [n]$ and $b \in \{0,1\}$:
5 : $\quad r_{i,b}, s_i \leftarrow \{0,1\}^\lambda$
6 : $\quad c_{i,b} \leftarrow \mathsf{WE.Enc}(1^\lambda, (p, i, b), r_{i,b})$
7 : $\quad k_{i,b} \leftarrow \mathsf{H}(r_{i,b} \,||\, 0) \oplus s_i$
8 : $\quad z_{i,b} \leftarrow \mathsf{H}(r_{i,b} \,||\, 1) \oplus X_{i,b}$
9 : $s \leftarrow s_1 \oplus \ldots \oplus s_n$
10 : $T \leftarrow \mathsf{H.Enc}^{\mathsf{H}}(s, \mathbf{z} \,||\, \widetilde{C} \,||\, d)$
11 : **Return** $\mathsf{sk}_p = (\mathbf{c}, \mathbf{k}, T)$

$\mathsf{FS.Sign}(\mathsf{sk}_p, m)$:

1 : If $p(m) \neq 1$: **Return** $\perp$
2 : Parse $m = (m_1, \ldots, m_n)$
3 : $r_{i,m_i} \leftarrow \mathsf{WE.Dec}(m, c_{i,m_i})$
4 : $s_i \leftarrow \mathsf{H}(r_{i,b} \,||\, 0) \oplus k_{i,m_i}$
5 : $s \leftarrow s_1 \oplus \ldots \oplus s_n$
6 : $(\mathbf{z} \,||\, \widetilde{C} \,||\, d) \leftarrow \mathsf{H.Dec}^{\mathsf{H}}(s, T)$
7 : $X_{i,m_i} \leftarrow \mathsf{H}(r_{i,m_i} \,||\, 1) \oplus z_{i,m_i}$
8 : $Y \leftarrow \mathsf{G.Ev}(\widetilde{C}, X)$
9 : $\sigma \leftarrow \mathsf{G.De}(d, Y)$
10 : **Return** σ

$\mathsf{FS.Vfy}(\mathsf{vk}, m, \sigma)$:

1 : **Return** $\mathsf{S.Vfy}(\mathsf{vk}, m, \sigma)$

Fig. 11. Functional Signature from WE and Garbling in the ROM. The WE scheme is for the relation $\mathcal{R}_{\mathsf{bit\text{-}sat}} = \{((p,i,b), w) : p(w) = 1 \ \wedge \ w_i = b\}$. As no CRS is used, we omit $\mathsf{FS.Setup}$.

Theorem 4. *If* WE *is a sound witness encryption for* $\mathcal{R}_{\mathsf{bit\text{-}sat}}$*,* G *a garbling scheme with (non-adaptive) simulation-based privacy, and* $(\mathsf{S.Gen}, \mathsf{S.Sign}, \mathsf{S.Vfy})$ *is an un unforgeable signature, then the scheme in Figure 11 is an unforgeable and one-time function-private functional signature for the class of predicates* $\mathcal{P}$ *with unique witness.*

Proof. We split the proof into showing unforgeability and function privacy.

Unforgeability. Let $\mathcal{A}$ be an adversary performing t key queries for predicates $p_1, \ldots, p_t$. We show through a sequence of hybrids $\mathsf{H}_0, \ldots, \mathsf{H}_5$ how to replace sk with oracle calls to $\mathsf{S.Sign}(\mathsf{sk}, \cdot)$ in the generation of sk_{p_i}, and only for messages m such that $p_i(m) = 1$. Our reduction heavily relies on the observability and programmability of the RO. In the following $x \succ y$ means the string y is a prefix of x. Finally let $x_1, \ldots, x_q$ be the RO queries performed by $\mathcal{A}$.

– H_0: The real key generation procedure for sk_p.

– H_1: Abort if at any point either Bad_1 or Bad_2 occurs where:
$\mathsf{Bad}_1 \;=\; \exists \alpha, \beta, i : (x_\alpha \succ r_{i,0} \;\wedge\; x_\beta \succ r_{i,1})$,
$\mathsf{Bad}_2 \;=\; \exists \alpha_1, \ldots, \alpha_n, b_1, \ldots, b_n : (p(b_1, \ldots, p_n) = 0 \;\wedge\; \forall i (x_{\alpha_i} \succ r_{i,b_i}))$.

– H_2: Abort if at any point Bad_3 occurs, where
$\mathsf{Bad}_3 \;=\; \exists \alpha \, (x_\alpha \succ s \;\wedge\; \exists i \, (\forall \beta < \alpha \;:\; x_\beta \not\succ r_{i,0} \;\wedge\; x_\beta \not\succ r_{i,1}))$.

– H_3: As H_2 but, when p is queried, sample $(\mathbf{r}, \mathbf{s}, s, \mathbf{c}, \mathbf{k})$ as in $\mathsf{FS.KeyGen}$ and $T \leftarrow \{0,1\}^\ell$. Return $\mathsf{sk}_p = (\mathbf{c}, \mathbf{k}, T)$. Next, when $\mathcal{A}$ queries $x_{\alpha_i} \succ r_{i,b_i}$ for all i, program H in points with prefix s so that $\mathsf{H.Dec}^{\mathsf{H}}(s, T) = \mathsf{Sim}_1(\mathsf{sk}, p, \mathbf{r}, \mathbf{b})$.

– H_4: As H_3 but replace $\mathsf{Sim}_1(\mathsf{sk}, p, \mathbf{r}, \mathbf{b})$ with $\mathsf{Sim}_2(\mathsf{sk}, p, \mathbf{r}, \mathbf{b})$.

– H_5: As H_4 but replace $\mathsf{Sim}_2(\mathsf{sk}, p, \mathbf{r}, \mathbf{b})$ with $\mathsf{Sim}_3^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(p, \mathbf{r}, \mathbf{b})$.

$\mathsf{Sim}_1(\mathsf{sk}, p, \mathbf{r}, \mathbf{b})$:

1 : Sample a random tape ρ
2 : $(\widetilde{C}, e, d) \leftarrow \mathsf{G.Gr}(\mathsf{S.Sign}(\mathsf{sk}, \cdot\,; \rho))$
3 : Parse $e = (X_{1,0}, \ldots, X_{n,1})$
4 : $z_{i,b} \leftarrow \mathsf{H}(r_{i,b} \,||\, 1) \oplus X_{i,b}$
5 : **Return** $(\mathbf{z}, \widetilde{C}, d)$

$\mathsf{Sim}_2(\mathsf{sk}, p, \mathbf{r}, \mathbf{b})$:

1 : Sample a random tape ρ
2 : $(\widetilde{C}, e, d) \leftarrow \mathsf{G.Gr}(\mathsf{S.Sign}(\mathsf{sk}, \cdot\,; \rho))$
3 : Parse $e = (X_{1,0}, \ldots, X_{n,1})$
4 : $z_{i,b_i} \leftarrow \mathsf{H}(r_{i,b_i} \,||\, 1) \oplus X_{i,b_i}$
5 : $z_{i,1-b_i} \leftarrow \{0,1\}^\lambda$
6 : **Return** $(\mathbf{z}, \widetilde{C}, d)$

$\mathsf{Sim}_3^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(p, \mathbf{r}, \mathbf{b})$:

1 : $\sigma \leftarrow \mathsf{S.Sign}(\mathsf{sk}, b_1 \,||\, \ldots \,||\, b_n)$
2 : $\left((X_{i,b_i})_{i=1}^n, \widetilde{C}, d\right) \leftarrow \mathcal{S}(1^\lambda, \sigma, \mathsf{S.Sign})$
3 : $z_{i,b_i} \leftarrow \mathsf{H}(r_{i,b_i} \,||\, 1) \oplus X_{i,b_i}$
4 : $z_{i,1-b_i} \leftarrow \{0,1\}^\lambda$
5 : **Return** $(\mathbf{z}, \widetilde{C}, d)$

Fig. 12. Sub-procedures used in hybrids H_3, H_4, H_5. We assume H to have output length λ. $\mathcal{S}$ is the garbling simulator from Definition 5.

$\mathsf{H}_0 \approx \mathsf{H}_1$. For any adversary $\mathcal{A}$ let $\mathcal{B}$ be the following adversary for the multiple-challenge WE game in Figure 2. Initially $\mathcal{B}(1^\lambda)$ samples $\mathsf{sk}, \mathsf{vk} \leftarrow \mathsf{FS.mskSetup}(1^\lambda)$ and answer all key generations queries normally except the one for p. When p is queried, $\mathcal{B}$ samples $r_{i,b} \leftarrow \{0,1\}^\lambda$, queries $((p, i, b), r_{i,b}, \mathbf{0})$ to its challenger to obtain $c_{i,b}$. Then in proceeds as $\mathsf{FS.KeyGen}$ computing $\mathsf{sk}_p = (\mathbf{c}, \mathbf{k}, T)$. Finally it returns 1 only if at the end of $\mathcal{A}$'s execution either Bad_1 or Bad_2 occurred.

If $\mathcal{B}$ is executed with challenge bit 0, then $c_{i,b} = \mathsf{WE.Enc}(1^\lambda, (p, i, b), r_{i,b})$. Thus it perfectly simulates H_0 and returns 1 with probability $\Pr[\mathsf{Bad}_1 \vee \mathsf{Bad}_2]$.

Conversely if its challenge bit is 1, the ciphertext $c_{i,b}$ encrypts $\mathbf{0}$ for all false instances (p,i,b). Let $\mathsf{Hit}_{i,b}$ be the probability $\mathcal{A}$ eventually queries $x \succ r_{i,b}$ for (p,i,b) false instance, and $\mathsf{Hit} = \vee_{i=1}^{n}(\mathsf{Hit}_{i,0} \vee \mathsf{Hit}_{i,1})$. We cannot directly argue $\mathcal{A}$ has no information on $r_{i,b}$ as it receives (a function of) $\mathsf{H}(r_{i,b} \,||\, 0)$, $\mathsf{H}(r_{i,b} \,||\, 1)$. However, until neither of those input is queried, those values are distributed uniformly and independently from $r_{i,b}$. Thus

$$\begin{aligned}\Pr\left[\mathsf{Hit}_{i,b}\right] \;&=\; \Pr\left[\nexists \alpha : x_\alpha \succ r_{i,b}\right] \;\leq\; \sum\nolimits_{\alpha=1}^{q} \Pr\left[x_\alpha \succ r_{i,b} \,|\, x_\beta \not\succ r_{i,b}\ \forall \beta < \alpha\right] \\ &\leq\; \sum\nolimits_{\alpha=1}^{q} \frac{1}{2^\lambda - \alpha + 1} \;\leq\; \frac{q}{2^\lambda - q}.\end{aligned}$$

A union bound yields that Hit occurs only with negligible probability $\frac{2nq}{2^\lambda - q}$. To conclude observe that $\mathsf{Bad}_1 \;\Rightarrow\; \mathsf{Hit}$ and $\mathsf{Bad}_2 \;\Rightarrow\; \mathsf{Hit}$. In the first case, since there exists by construction only a unique witness for p, and in the second since if by contradiction (p,i,b_i) were all true, then the existence of a unique witness implies $p(b_1,\ldots,b_n) = 1$. We thus conclude that $\mathcal{B}$ return 1 in this case only with negligible probability. Hence

$$\Pr\left[\mathsf{Bad}_1 \vee \mathsf{Bad}_2 \,|\, \mathsf{H}_0\right] \;\leq\; \mathsf{Adv}_{\mathcal{B}}(\lambda) + \mathsf{negl}(\lambda) \;=\; \mathsf{negl}(\lambda).$$

In particular, the advantage of any distinguisher for $\mathsf{H}_0, \mathsf{H}_1$ is negligible as the two worlds are identical when $\neg(\mathsf{Bad}_1 \vee \mathsf{Bad}_2)$.

$\mathsf{H}_1 \approx \mathsf{H}_2$. It suffices to show that Bad_3 occurs with negligible probability in H_0. Once the adversary $\mathcal{A}$ queries sk_p, all the information it receives about s is contained in T, which is a function of $\mathsf{H}(s \,||\, 0)$, $\mathsf{H}(s \,||\, 1)$, ... and in $k_{i,b} = \mathsf{H}(r_{i,b} \,||\, 0) \oplus s_i$. Until a string with prefix s is queried, T is independently distributed from s. Moreover, until either $r_{i,0}$ or $r_{i,1}$ is queries, s_i is uniformly distributed, and in particular so is $s = s_1 \oplus \ldots \oplus s_n$ conditioning on $\mathcal{A}$'s view. Therefore, calling $\mathsf{Miss}_{\alpha,i}$ the event in which $r_{i,b}$ is never queried before the α-th query, we have that

$$\begin{aligned}\Pr\left[\mathsf{Bad}_3\right] \;&=\; \Pr\left[\exists \alpha : x_\alpha \succ s \;\wedge\; \mathsf{Miss}_{\alpha,i}\right] \\ &\leq\; \sum\nolimits_{\alpha=1}^{q} \Pr\left[x_\alpha \succ s \;\wedge\; \mathsf{Miss}_{\alpha,i} \,|\, x_\beta \not\succ s\ \forall \beta < \alpha\right] \\ &\leq\; \sum\nolimits_{\alpha=1}^{q} \frac{1}{2^\lambda - \alpha + 1} \;\leq\; \frac{q}{2^\lambda - q} \;=\; \mathsf{negl}(\lambda).\end{aligned}$$

$\mathsf{H}_2 \approx \mathsf{H}_3$. First, the programming succeeds as $\neg\mathsf{Bad}_3$ implies no point with prefix s was queried yet, or else the hybrid just aborts. Next, as T is uniformly sampled, the programmed values $(\mathsf{H}(s \,||\, 0), \ldots)$ follow the same distribution as in H_2. Finally, since in both games $\mathsf{H}.\mathsf{Dec}^{\mathsf{H}}(s,T) = (\mathbf{z}, \widetilde{C}, d)$ with the latter being identically distributed in the two hybrids, the two games are perfectly indistinguishable.

$\mathsf{H}_3 \approx \mathsf{H}_4$. Since $\neg\mathsf{Bad}_1$, or else both hybrids simply aborts, we have that once $\mathcal{A}$ queries input with prefixes r_{i,b_i}, then it also never queries any string with prefix $r_{i,1-b_i}$. It follows that $\mathsf{H}(r_{i,1-b_i} \,||\, 1) \oplus X_{i,1-b_i}$ and $z_{i,1-b_i}$ (as per H_4) are identically distributed conditioning on $\mathcal{A}$'s view. Again, the two hybrids are thus perfectly indistinguishable.

$\mathsf{H}_4 \approx \mathsf{H}_5$. Let $\mathcal{D}$ be a distinguisher for the two hybrids. We define $\mathcal{B}$ against the garbling privacy, with leakage function Φ revealing only the circuit topology. $\mathcal{B}(1^\lambda)$ executes H_4 correctly for all key requests except p. In that case, as in H_4 it returns $(\mathbf{c}, \mathbf{k}, T)$. When $\mathcal{D}$ queries $x_{\alpha_i} \succ r_{i,b_i}$, it programs the output of $\mathsf{H.Dec}^{\mathsf{H}}(s, T)$ as follows. First, it sets $m = (b_1, \ldots, b_n)$ and sends $(\mathsf{S.Sign}(\mathsf{sk}, \cdot; \rho), m)$ (for a randomly sampled tape ρ) to its challenger for garbling, and waits for its reply $(\widetilde{C}, X, d)$. It then sets $z_{i,b_i} = \mathsf{H}(r_{i,b} \,||\, X_i)$, samples $z_{i,1-b_i}$ uniformly and programs $\mathsf{H.Dec}^{\mathsf{H}}(s, T) = (\mathbf{z}, \widetilde{C}, d)$. Finally, once $\mathcal{D}$ returns a bit, $\mathcal{B}$ returns the same.

When $\mathcal{B}$ has challenge bit 0, then its challenger computed $(\widetilde{C}_0, e_0, d_0)$ as a correct garbling of $\mathsf{S.Sign}(\mathsf{sk}, \cdot; \rho)$, and set $X = (X_{i,b_i})$ for $e_0 = (X_{1,0}, \ldots, X_{n,1})$. Thus in this case $\mathcal{B}$ perfectly simulates H_4. Similarly, when its challenge bit is 1, $X, \widetilde{C}, d$ is computed by $\mathcal{S}$ on output value $\sigma = \mathsf{S.Sign}(\mathsf{sk}, m)$ and circuit information $\Phi(\mathsf{S.Sign}(\mathsf{sk}, \cdot; \rho) = \mathsf{S.Sign}$. Thus it perfectly simulates H_5. Hence $\mathsf{Adv}_{\mathcal{D}}(\lambda) = \mathsf{Adv}_{\mathcal{B}}(\lambda) = \mathsf{negl}(\lambda)$.

Conclusion. Applying the hybrids above to each signature query performed by $\mathcal{A}$, let H^* be the resulting hybrid. In such game all key generation queries for sk_{p_i} only requires oracle calls to $\mathsf{S.Sign}(\mathsf{sk}, m)$ for some m such that $p_i(m) = 1$. Let then $\mathcal{B}$ be the following adversary for the underlying signature scheme's unforgeability. $\mathcal{B}(1^\lambda, \mathsf{vk})$ simulates H^* with $\mathcal{A}$, forwarding the signing queries of H^* to its challenger. When $\mathcal{A}$ returns (m^*, σ^*), $\mathcal{B}$ returns the same.

Assume (m^*, σ^*) is a valid forgery for the functional signature scheme, i.e. that m^* was never sent to $\mathcal{A}'s$ signing oracle, and $p_i(m^*) = 0$ for all p_i. Then H^* (and so $\mathcal{B}$) also never queried $\mathsf{S.Sign}(\mathsf{sk}, m^*)$. Thus (m^*, σ^*) is a valid forgery for the underlying signature scheme. We can then conclude that

$$\mathsf{Adv}_{\mathcal{A}}(\lambda) \;=\; \mathsf{negl}(\lambda) + \mathsf{Adv}_{\mathcal{B}}(\lambda) \;=\; \mathsf{negl}(\lambda).$$

One-Time Function Privacy. Due to garble circuit correctness, given vk, sk and p, m such that $p(m) = 1$, then $\sigma_0 = \mathsf{FS.Sign}(\mathsf{sk}_p, m) = \mathsf{S.Sign}(\mathsf{sk}, m; \rho)$ with ρ being the random tape sampled by $\mathsf{FS.KeyGen}$ while computing sk_p. Since ρ is distributed uniformly and independently from vk, sk, it follows that σ_0 follows the same distribution of $\sigma_1 \leftarrow \mathsf{S.Sign}(\mathsf{sk}, m)$ conditioning on vk, sk. In particular one-time function privacy follows information-theoretically.

5.3 Adding Key Verifiability

A major limitation of the scheme in Section 5.2 is that it is not directly compatible with the compiler in Section 4. Specifically, showing a functional key sk_p to

be correct would require to prove statements about the RO, which is not possible only relative to H.

Common strategies to overcome this issue includes either augmenting the RO with extra functionality (e.g. the signed ROM in [CT10]), or by assuming that security is preserved when the RO is instantiated with a concrete hash function. For the specific case of our construction it can be shown that the pairing-based Generic Group Model [Sho97, BB04] suffices for our goal. Indeed, in this model NIZK about *algebraic* relations, which informally can be verified by a set of linear equations in $\mathbb{G}_1$ and quadratic relations in $\mathbb{Z}_p$, exists[9] [GS08].

Up to minor adaptations, which we discuss in greater details in the Appendix, Section B.2, the key generation procedure can in fact be expressed through an algebraic relation, in the sense of [GS08]. At a high level, this is simply achieved replacing the RO usage as a symmetric key encryption scheme with (bit wise) El-Gamal in the exponent.

References

ACdMT05. Giuseppe Ateniese, Daniel H. Chou, Breno de Medeiros, and Gene Tsudik. Sanitizable signatures. In Sabrina De Capitani di Vimercati, Paul F. Syverson, and Dieter Gollmann, editors, *ESORICS 2005*, volume 3679 of *LNCS*, pages 159–177. Springer, Berlin, Heidelberg, September 2005.

AEE+21. Lukas Aumayr, Oguzhan Ersoy, Andreas Erwig, Sebastian Faust, Kristina Hostáková, Matteo Maffei, Pedro Moreno-Sanchez, and Siavash Riahi. Generalized channels from limited blockchain scripts and adaptor signatures. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part II*, volume 13091 of *LNCS*, pages 635–664. Springer, Cham, December 2021.

AME+21. Lukas Aumayr, Matteo Maffei, Oguzhan Ersoy, Andreas Erwig, Sebastian Faust, Siavash Riahi, Kristina Hostáková, and Pedro Moreno-Sanchez. Bitcoin-compatible virtual channels. In *2021 IEEE Symposium on Security and Privacy*, pages 901–918. IEEE Computer Society Press, May 2021.

BB04. Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 56–73. Springer, Berlin, Heidelberg, May 2004.

BBS04. Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 41–55. Springer, Berlin, Heidelberg, August 2004.

BF14. Mihir Bellare and Georg Fuchsbauer. Policy-based signatures. In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 520–537. Springer, Berlin, Heidelberg, March 2014.

[9] Arguably the same class of proofs exist in the *plain* GGM by combining any standard model NIZK for NP and a generalized Schnorr proof [Cra96] where the RO is instantiated as in [ZZ21]. However, on the one hand such use of the GGM has been historically criticized [KM06], while on the other hand restricting to purely algebraic usage of the GGM means such proofs are again impossible [Giu23].

BGI+01. Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, Berlin, Heidelberg, August 2001.

BGI14. Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 501–519. Springer, Berlin, Heidelberg, March 2014.

BHR12. Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *ACM CCS 2012*, pages 784–796. ACM Press, October 2012.

BKV19. Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 227–247. Springer, Cham, December 2019.

BMS16. Michael Backes, Sebastian Meiser, and Dominique Schröder. Delegatable functional signatures. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 357–386. Springer, Berlin, Heidelberg, March 2016.

BSW06. Dan Boneh, Emily Shen, and Brent Waters. Strongly unforgeable signatures based on computational Diffie-Hellman. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *PKC 2006*, volume 3958 of *LNCS*, pages 229–240. Springer, Berlin, Heidelberg, April 2006.

Cha83. David Chaum. Blind signature system. In David Chaum, editor, *CRYPTO'83*, page 153. Plenum Press, New York, USA, 1983.

CL03. Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In Stelvio Cimato, Clemente Galdi, and Giuseppe Persiano, editors, *SCN 02*, volume 2576 of *LNCS*, pages 268–289. Springer, Berlin, Heidelberg, September 2003.

Cra96. Ronald Cramer. Modular design of secure yet practical cryptographic protocols. *Ph. D.-thesis, CWI and U. of Amsterdam*, 2, 1996.

CT10. Alessandro Chiesa and Eran Tromer. Proof-carrying data and hearsay arguments from signature cards. In *ICS*, volume 10, pages 310–331, 2010.

DH76. Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

DKL+18. Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: A lattice-based digital signature scheme. *IACR TCHES*, 2018(1):238–268, 2018.

DOY22. Wei Dai, Tatsuaki Okamoto, and Go Yamamoto. Stronger security and generic constructions for adaptor signatures. In Takanori Isobe and Santanu Sarkar, editors, *INDOCRYPT 2022*, volume 13774 of *LNCS*, pages 52–77. Springer, Cham, December 2022.

DST18. Thomas Debris-Alazard, Nicolas Sendrier, and Jean-Pierre Tillich. Wave: A new family of trapdoor one-way preimage sampleable functions based on codes. Cryptology ePrint Archive, Report 2018/996, 2018.

EEE20. Muhammed F. Esgin, Oguzhan Ersoy, and Zekeriya Erkin. Post-quantum adaptor signatures and payment channel networks. In Liqun Chen, Ninghui Li, Kaitai Liang, and Steve A. Schneider, editors, *ESORICS 2020, Part II*, volume 12309 of *LNCS*, pages 378–397. Springer, Cham, September 2020.

EFH+21. Andreas Erwig, Sebastian Faust, Kristina Hostáková, Monosij Maitra, and Siavash Riahi. Two-party adaptor signatures from identification schemes. In Juan Garay, editor, *PKC 2021, Part I*, volume 12710 of *LNCS*, pages 451–480. Springer, Cham, May 2021.

Fis05. Marc Fischlin. Communication-efficient non-interactive proofs of knowledge with online extractors. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 152–168. Springer, Berlin, Heidelberg, August 2005.

GGSW13. Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 467–476. ACM Press, June 2013.

Giu23. Emanuele Giunta. On the impossibility of algebraic NIZK in pairing-free groups. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part IV*, volume 14084 of *LNCS*, pages 702–730. Springer, Cham, August 2023.

GMM+22. Noemi Glaeser, Matteo Maffei, Giulio Malavolta, Pedro Moreno-Sanchez, Erkan Tairi, and Sri Aravinda Krishnan Thyagarajan. Foundations of coin mixing services. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 1259–1273. ACM Press, November 2022.

GMR88. Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, April 1988.

GQ90. Louis C. Guillou and Jean-Jacques Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In Shafi Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 216–231. Springer, New York, August 1990.

GS08. Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *LNCS*, pages 415–432. Springer, Berlin, Heidelberg, April 2008.

GSST24. Paul Gerhart, Dominique Schröder, Pratik Soni, and Sri Aravinda Krishnan Thyagarajan. Foundations of adaptor signatures. In Marc Joye and Gregor Leander, editors, *EUROCRYPT 2024, Part II*, volume 14652 of *LNCS*, pages 161–189. Springer, Cham, May 2024.

HJK+16. Dennis Hofheinz, Tibor Jager, Dakshita Khurana, Amit Sahai, Brent Waters, and Mark Zhandry. How to generate and use universal samplers. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 715–744. Springer, Berlin, Heidelberg, December 2016.

KH22. Jean Belo Klamti and M. Anwar Hasan. Post-quantum two-party adaptor signature based on coding theory. *Cryptogr.*, 6(1):6, 2022.

KM06. Neal Koblitz and Alfred Menezes. Another look at generic groups. Cryptology ePrint Archive, Report 2006/230, 2006.

KW03. Jonathan Katz and Nan Wang. Efficiency improvements for signature schemes with tight security reductions. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *ACM CCS 2003*, pages 155–164. ACM Press, October 2003.

LIZ24. Xiangyu Liu, Tzannetos Ioannis, and Vassilis Zikas. Adaptor signatures: New security definition and a generic construction for np relations. *Cryptology ePrint Archive*, 2024.

MMS+19. Giulio Malavolta, Pedro Moreno-Sanchez, Clara Schneidewind, Aniket Kate, and Matteo Maffei. Anonymous multi-hop locks for blockchain scalability and interoperability. In *NDSS 2019*. The Internet Society, February 2019.

MTV+23. Varun Madathil, Sri Aravinda Krishnan Thyagarajan, Dimitrios Vasilopoulos, Lloyd Fournier, Giulio Malavolta, and Pedro Moreno-Sanchez. Cryptographic oracle-based conditional payments. In *NDSS 2023*. The Internet Society, February 2023.

OPR. Bitcoin wiki, page op_return. `https://en.bitcoin.it/wiki/OP_RETURN`.

Poe17. Andrew Poelstra. Lightning in scriptless scripts. `https://tinyurl.com/mcefmph`, 2017.

Poe18. Andrew Poelstra. Mimblewimble and scriptless scripts. presentation slides (1 2018). `https://download.wpsoftware.net/bitcoin/wizardry/mw-slides/2018-01-10-rwc/slides.pdf`, 2018.

QPM+23. Xianrui Qin, Shimin Pan, Arash Mirzaei, Zhimei Sui, Oguzhan Ersoy, Amin Sakzad, Muhammed F. Esgin, Joseph K. Liu, Jiangshan Yu, and Tsz Hon Yuen. BlindHub: Bitcoin-compatible privacy-preserving payment channel hubs supporting variable amounts. In *2023 IEEE Symposium on Security and Privacy*, pages 2462–2480. IEEE Computer Society Press, May 2023.

RSA78. Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, February 1978.

Sch91. Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, January 1991.

Sha84. Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 47–53. Springer, Berlin, Heidelberg, August 1984.

Sho97. Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 256–266. Springer, Berlin, Heidelberg, May 1997.

SW14. Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014.

TMM21. Erkan Tairi, Pedro Moreno-Sanchez, and Matteo Maffei. Post-quantum adaptor signature for privacy-preserving off-chain payments. In Nikita Borisov and Claudia Díaz, editors, *FC 2021, Part II*, volume 12675 of *LNCS*, pages 131–150. Springer, Berlin, Heidelberg, March 2021.

TMM22. Sri Aravinda Krishnan Thyagarajan, Giulio Malavolta, and Pedro Moreno-Sanchez. Universal atomic swaps: Secure exchange of coins across all blockchains. In *2022 IEEE Symposium on Security and Privacy*, pages 1299–1316. IEEE Computer Society Press, May 2022.

TZY22. Binbin Tu, Min Zhang, and Chen Yu. Efficient ECDSA-based adaptor signature for batched atomic swaps. In Willy Susilo, Xiaofeng Chen, Fuchun Guo, Yudi Zhang, and Rolly Intan, editors, *ISC 2022*, volume 13640 of *LNCS*, pages 175–193. Springer, Cham, December 2022.

Yao82. Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd FOCS*, pages 160–164. IEEE Computer Society Press, November 1982.

ZZ21. Mark Zhandry and Cong Zhang. The relationship between idealized models under computationally bounded adversaries. Cryptology ePrint Archive, Report 2021/240, 2021.

A Examples of Extractable Predicates

In this section we discuss possible instantiations of extractable predicates with *unique witness* (Definition 15). Specifically we realize them directly from the preimage relation of trapdoor permutation, or by combining *any* unique witness predicate with an argument of knowledge.

Let us note that, to the best of our knowledge, AS w.r.t. relations with unique witnesses is sufficient for all existing applications of AS. And indeed, most AS construction from concrete signature schemes (which are also the one most likely AS constructions used in practice due to their efficiency) are defined w.r.t. some fixed relation in UP (i.e., NP relation with unique witnesses).

A.1 From Trapdoor Permutation

Let us recall notation for trapdoor permutations, originally introduced in [DH76, RSA78]. A keyed function family $f_k : X \to X$ is a trapdoor permutation if there exists $(\mathsf{Gen}, \mathsf{Inv})$ such that: $(k, \mathsf{td}) \leftarrow \mathsf{Gen}(1^\lambda)$ generates the evaluation key along with trapdoor td; the function $f_k : X \to X$ is a (keyed) one way-permutation; $\mathsf{Inv}(\mathsf{td}, y) \to x$ such that $f_k(x) = y$ for any y in the image of f_k. Given any trapdoor permutation $(\mathsf{Gen}, \mathsf{Inv})$, a family of extractable predicates $(\mathcal{P}, \mathsf{R.Gen}, \mathsf{R.Ext})$ is readily derived as follows:

- $\mathcal{P}$ consists of all the predicates $p_y(k, x) : (f_k(x) = y)$.
- $\mathsf{R.Gen}(1^\lambda)$ samples $(k, \mathsf{td}) \leftarrow \mathsf{Gen}(1^\lambda)$ and returns $(\mathsf{crs}, \mathsf{td})$ with $\mathsf{crs} = k$.
- $\mathsf{R.Ext}(\mathsf{td}, p_y)$ returns $\mathsf{Inv}(\mathsf{td}, y)$.

This satisfies Definition 15. Indeed sampling (k, td), a predicate $p_y(k, \cdot)$ is satisfiable if and only if $y \in \mathrm{Im}\, f_k$. Hence, due to the correctness of Inv, the extraction always succeeds. Unique witness follows from f_k being a permutation for all k.

A.2 From Non-Interactive Arguments of Knowledge

Another way to instantiate extractable predicates is through NIZK argument of knowledge (NIZK-AoK). We explain this approach for NIZK-AoK in the CRS model first as it fits our definition of extractable predicates better. Later, we move to NIZK-AoK in the ROM, which will require careful considerations when applied to both our constructions in Section 5. Let us note that the latter approach has been taken by prior works on adaptor signatures; namely, by Erwig et al. [AEE+21] for ECDSA and Tairi et al. [TMM21] for CSI-FiSh.

Construction in the CRS model. Let $\mathcal{Q}$ be a family of predicates with unique witness. Let $(\mathsf{NIZK.G}, \mathsf{NIZK.P}, \mathsf{NIZK.V})$ be a NIZK-AoK (e.g. [GS08]) for predicate satisfiability, defined as

$$\mathcal{R}_{\mathsf{sat},\mathcal{Q}} = \{(q, m) : q \in \mathcal{Q} \wedge q(m) = 1\}.$$

Since we assume NIZK to be an arugment of knowledge, there exists an extractor $\mathcal{E} = (\mathcal{E}_1, \mathcal{E}_2)$ such that $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathcal{E}_1(1^\lambda)$ and $m \leftarrow \mathcal{E}_2(\mathsf{td}, q, \pi)$ extracts a witness with overwhelming probability given an accepting proof for $q \in \mathcal{Q}$. We can then augment $\mathcal{Q}$ to an extractable relation $(\mathcal{P}, \mathsf{R.Gen}, \mathsf{R.Ext})$ as follows:

- $\mathcal{P}$ is the set of predicates $p_{q,\pi}(\mathsf{crs}, m) : (q(m) = 1) \wedge (\mathsf{NIZK.V}(\mathsf{crs}, q, \pi) = 1)$,
- $\mathsf{R.Gen}(1^\lambda)$ returns $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathcal{E}_1(1^\lambda)$, and
- $\mathsf{R.Ext}(\mathsf{td}, p_{q,\pi})$ returns $\perp$ if $\mathsf{NIZK.V}(\mathsf{crs}, q, \pi) = 0$, otherwise $m \leftarrow \mathcal{E}_2(\mathsf{td}, q, \pi)$.

Correctness of the triplet follows since any satisfiable $p_{q,\pi}$ must have an accepting π. In particular q is satisfiable and $\mathcal{E}_2(\mathsf{td}, q, \pi)$ returns a valid assignment for q with overwhelming probability. Finally unique witness is directly inherited from $\mathcal{Q}$.

Construction in the RO model. In order to extend the notion of extractable relations to the RO, i.e. without any common reference string, we need to introduce the following syntactical changes. First of all, that $\perp \leftarrow \mathsf{R.Gen}^{\mathsf{H}}(1^\lambda)$, i.e. no crs/td is ever generated. Secondly, we need to assume that $\mathsf{R.Ext}^{\mathsf{H}_{\mathsf{ob}}}$ can *observe* all previous RO queries, where we formally denote with H_{ob} an oracle returning the query history of H.

Given these changes, let $\mathcal{Q}$ be a family of predicates with unique witness, and $(\mathsf{NIZK.P}^{\mathsf{H}}, \mathsf{NIZK.V}^{\mathsf{H}})$ a straight-line extractable NIZK for $\mathcal{R}_{\mathsf{sat},\mathcal{Q}}$ (this can be constructed using Fischlin transform [Fis05]). By *straight-line* extractable here we mean that there exists an extractor $\mathcal{E}^{\mathsf{H}_{\mathsf{ob}}}$ such that, if $\mathsf{NIZK.V}^{\mathsf{H}}(q, \pi)$ accepts, then $m \leftarrow \mathcal{E}^{\mathsf{H}_{\mathsf{ob}}}(q, \pi)$ where, with overwhelming probability, $q(m) = 1$. An extractable relation can then be built defining

- $\mathcal{P}$ the set of predicates $p_{q,\pi}(m) : (q(m) = 1) \wedge (\mathsf{NIZK.V}^{\mathsf{H}}(q, \pi) = 1)$.
- $\mathsf{R.Ext}^{\mathsf{H}_{\mathsf{ob}}}(p_{q,\pi})$ returns $\perp$ if $\mathsf{NIZK.V}^{\mathsf{H}}(q, \pi) = 0$, otherwise $m \leftarrow \mathcal{E}^{\mathsf{H}_{\mathsf{ob}}}(q, \pi)$.

We finally stress that in the obfuscation-based construction we provide in Section 5.1, such extractable predicate would not be applicable directly. Indeed, there we need the obfuscation of a circuit $C_{\mathsf{sk},p,r}(m)$ which returns $\mathsf{S.Sign}(\mathsf{sk}, m; r)$ if $p(m) = 1$. However in this case p *technically* depends on the H due to the clause involving $\mathsf{NIZK.V}^{\mathsf{H}}(\cdot, \cdot)$. Nevertheless, such dependency is not an issue. Indeed $\mathsf{NIZK.V}^{\mathsf{H}}(q, \pi)$ does not depend on m – the only free variable in p – and is therefore a constant. An explicit circuit for $C_{\mathsf{sk},p,r}$ can then still be obtained by pre-evaluating the bit $\mathsf{NIZK.V}^{\mathsf{H}}(q, \pi)$ beforehand and hard-coding it into the circuit.

B Replacing the ROM with the GGM

B.1 Generic Group Model

Here we recall notation for the bilinear generic group model (adapted from [Sho97]). Given a prime $p = \Omega(2^\lambda)$, let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t$ be three copies of $\mathbb{Z}_p$ and L a random labeling function such that

$$L : \mathbb{G}_1 \sqcup \mathbb{G}_2 \sqcup \mathbb{G}_t \to \{0,1\}^\ell.$$

Where $\ell > \log p$.[10] The GGM is then defined by three oracles $\mathcal{O}_{\mathsf{gen}}, \mathcal{O}_{\mathsf{add}}, \mathcal{O}_{\mathsf{bil}}$, respectively returning canonical generators for $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t$, performing addition within a given group, and computing a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$.

B.2 Alternative Construction from WE and Garbling

In this section an alternative realization the almost scriptless FS from WE and garbling is presented. In the following we denote $(\mathsf{Enc}^{\mathsf{G}}, \mathsf{Dec}^{\mathsf{G}})$ a symmetric variants of ElGamal *in the exponent*. Formally, for message $m \in \{0,1\}$

- $\mathsf{Enc}^{\mathsf{G}}(r, m)$ samples $H \leftarrow \mathbb{G}_1$ and returns $(H, rH + mG)$
- $\mathsf{Dec}^{\mathsf{G}}(r, C_1, C_2)$ return $m \in \{0,1\}$ if $C_2 - rC_1 = mG$.

For arbitrary bit strings $m \in \{0,1\}^*$, we interpret $\mathsf{Enc}^{\mathsf{G}}(r, m)$ as being applied bit-wise. The full scheme is presented in Figure 13.

Theorem 5. *Relative to the Bilinear GGM oracles, there exists an adaptively sound NIZK* $(\mathsf{NIZK.G}, \mathsf{NIZK.P}, \mathsf{NIZK.V})$ *for the relation* $\mathcal{R}_{\mathsf{key}}$ *defined as*

$$\mathcal{R}_{\mathsf{key}} = \left\{ ((\mathsf{vk}, p, \mathsf{sk}_p), (\mathsf{sk}, R)) \ : \ \begin{array}{l} (\mathsf{vk}, \mathsf{sk}) = \mathsf{FS.mskSetup}(1^\lambda; \mathsf{sk}) \\ \mathsf{sk}_p = \mathsf{FS.KeyGen}(\mathsf{sk}, p; R) \end{array} \right\}$$

for the scheme FS *described in Figure 13.*

Proof. Since Groth-Sahai proofs [GS08] can be instantiated relative to the Bilinear GGM, we only have to show $\mathcal{R}_{\mathsf{key}}$ can be expressed through their target relation.

The first condition, $(\mathsf{vk}, \mathsf{sk})$ in the support of $\mathsf{FS.mskSetup}(1^\lambda)$, does not involve any group operation and can therefore be simply compiled to a set of quadratic constraints. Regarding the second part, the statement is equivalent to showing the existence of $r_{i,j}, s_i, R^0_{i,b}, R^1_{i,b}$ and random tapes $\mathsf{we}_{i,b}$ and grb such that

1. $c_{i,b} = \mathsf{WE.Enc}(1^\lambda, (p, i, b), r_{i,b}; \mathsf{we}_{i,b})$
2. $(s_i, R^0_{i,b}) = \mathsf{Dec}^{\mathsf{G}}(r_{i,b}, k_{i,b})$

[10] In general, whether $\ell = \log p + O(\log \lambda)$ or $\ell = \log p + \omega(\log \lambda)$ makes a significant difference, as oblivious sampling of group elements is only efficient in the first case. Such difference will however be irrelevant in our construction.

$\mathsf{FS.mskSetup}(1^\lambda)$:

$\mathsf{sk}, \mathsf{vk} \leftarrow \mathsf{S.Gen}(1^\lambda)$

Return $(\mathsf{sk}, \mathsf{vk})$

$\mathsf{FS.KeyGen}(\mathsf{sk}, p)$:

Sample a random tape ρ

$(\widetilde{C}, e, d) \leftarrow \mathsf{G.Gr}(\mathsf{S.Sign}(\mathsf{sk}, \cdot\,; \rho))$

Parse $e = (X_{1,0}, \ldots, X_{n,1})$

// 2-additive sharing of each $X_{i,b}$:

Sample $R^0_{i,b} \leftarrow \{0,1\}^\lambda$

Set $R^1_{i,b} = X_{i,b} \oplus R^0_{i,b}$

For $i \in [n]$ and $b \in \{0,1\}$:

  $r_{i,b}, s_i \leftarrow \{0,1\}^\lambda$

  $c_{i,b} \leftarrow \mathsf{WE.Enc}(1^\lambda, (p, i, b), r_{i,b})$

  $k_{i,b} \leftarrow \mathsf{Enc}^{\mathsf{G}}(r_{i,b}, (s_i, R^0_{i,b}))$

$s \leftarrow s_1 \oplus \ldots \oplus s_n$

$T \leftarrow \mathsf{Enc}^{\mathsf{G}}(s, \mathbf{R}^1 \,||\, \widetilde{C} \,||\, d)$

Return $\mathsf{sk}_p = (\mathbf{c}, \mathbf{k}, T)$

$\mathsf{FS.Sign}(\mathsf{sk}_p, m)$:

If $p(m) \neq 1$: **Return** $\perp$

Parse $m = (m_1, \ldots, m_n)$

$r_{i,m_i} \leftarrow \mathsf{WE.Dec}(m, c_{i,m_i})$

// Recover s and decrypt T

$s_i \leftarrow \mathsf{Dec}^{\mathsf{G}}(r_{i,m_i}, k_{i,m_i})$

$s \leftarrow s_1 \oplus \ldots \oplus s_n$

$(\mathbf{R}^1 \,||\, \widetilde{C} \,||\, d) \leftarrow \mathsf{Dec}^{\mathsf{G}}(s, T)$

// Reconstruct input labels X_{i,m_i}

$R^0_{i,m_i} \leftarrow \mathsf{Dec}^{\mathsf{G}}(r_{i,m_i}, z_{i,m_i})$

$X_{i,m_i} \leftarrow R^0_{i,m_i} \oplus R^1_{i,m_i}$

// Evaluate the garbled circuit

$Y \leftarrow \mathsf{G.Ev}(\widetilde{C}, X)$

$\sigma \leftarrow \mathsf{G.De}(d, Y)$

Return σ

$\mathsf{FS.Vfy}(\mathsf{vk}, m, \sigma)$:

Return $\mathsf{S.Vfy}(\mathsf{vk}, m, \sigma)$

Fig. 13. Functional Signature from WE and Garbling in the GGM. For simplicity we assume $X_{i,b} \in \{0,1\}^\lambda$.

3. $(\mathbf{R}^1, \widetilde{C}, d) = \mathsf{Dec}^{\mathsf{G}}(s, T)$
4. $e = (R^0_{i,b} \oplus R^1_{i,b})$
5. $(\widetilde{C}, e, d) = \mathsf{G.Gr}(\mathsf{S.Sign}(\mathsf{sk}, \cdot; \rho); \mathsf{grb})$

The first and last two statements are again independent from GGM operations and can therefore be expressed as a set of quadratic constraints. Conversely 2 and 3 can both be compiled as linear constraints over $\mathbb{G}_1$, since each individual $\mathsf{Dec}^{\mathsf{G}}$ term can be expressed in such way. This completes the proof.

Theorem 6. *If* WE *is a sound witness encryption for* $\mathcal{R}_{\mathsf{bit\text{-}sat}}$*,* G *a garbling scheme with simulation-based privacy, and* $(\mathsf{S.Gen}, \mathsf{S.Sign}, \mathsf{S.Vfy})$ *an unforgeable signature, then the scheme in Figure 13 is an unforgeable and one-time function private functional signature in the Bilinear GGM, for the class of predicates* $\mathcal{P}$ *with unique witness.*

Proof. The proof of one-time function privacy is identical to the one in Theorem 4. Regarding the proof of unforgeability, we follow the same structure of Theorem 4 up to minor adaptations. First, we observe that any reduction simulating the GGM and providing group elements $G, G_1, \ldots, G_\ell$ to the adversary, (including those $\mathcal{A}$ randomly sampled) can locally represent any group element $\mathcal{A}$ computes as a linear combination of them. In particular this gives rise to a set

of linear equations when two group element (computed in different ways) have the same representation, and non-equality otherwise.

Given $c = (H, C) = \mathsf{Enc}^{\mathsf{G}}(r, b)$ we define the event $\mathsf{dc}(\alpha, c)$ if, given the first α GGM queries and r it is possible to compute b. Note $\mathsf{dc}(\alpha, c)$ can be decided in polynomial time, as it involves checking whether the set of equations described above implies either $H^r C^{-1} = 0$ or $H^r C^{-1} \neq 0$. More generally, given $c_1, \ldots c_\ell$ we define $\mathsf{dc}(\alpha, c_1, \ldots, c_\ell) = \vee_{i=1}^{\ell} \mathsf{dc}(\alpha, c_i)$. With the above definition we proceed defining 6 hybrids $\mathsf{H}_0, \ldots, \mathsf{H}_5$ meant to replace the key generation procedure for the j-th key request for predicate p, removing the dependency on sk.

- H_0: The real generation procedure for sk_p.
- H_1: Abort if either Bad_1 or Bad_2 occurs, where:
  $\mathsf{Bad}_1 = \exists \alpha, \beta, i : (\mathsf{dc}(\alpha, k_{i,0}) \wedge \mathsf{dc}(\alpha, k_{i,1}))$
  $\mathsf{Bad}_2 = \alpha_1, \ldots, \alpha_n, b_1, \ldots, b_n : (p(b_1, \ldots, b_n) = 0 \ \wedge \ \forall i (\mathsf{dc}(\alpha_i, k_{i,b_i})))$
- H_2: As H_1, but abort if Bad_3 occurs, where
  $\mathsf{Bad}_3 = \exists \alpha \, (\mathsf{dc}(\alpha, T) \ \wedge \ \exists i \, (\forall \beta < \alpha \ : \ \mathsf{dc}(\beta, k_{i,0}) \ \wedge \ \mathsf{dc}(\beta, k_{i,1})))$
- H_3: As H_2, but when p is queries sample $(\mathbf{r}, \mathbf{s}, \mathbf{R}^0, s, \mathbf{c}, \mathbf{k})$ as in $\mathsf{FS.KeyGen}$ and $T \leftarrow \{0,1\}^\ell$. Return $\mathsf{sk}_p = (\mathbf{c}, \mathbf{k}, T)$. Next, when $\mathcal{A}$ performs GGM queries such that $\mathsf{dc}(\alpha_i, k_{i,b_i})$ for all $i \in [n]$, program the GGM so that $\mathsf{Dec}^{\mathsf{G}}(s, T) = \mathsf{Sim}_1(\mathsf{sk}, \mathbf{r}, \mathbf{R}^0, \mathbf{b})$, see Figure 14.
- H_4: As H_3, but replace Sim_1 with Sim_2.
- H_5: As H_4, but replace Sim_2 with Sim_3.

$\mathsf{Sim}_1(\mathsf{sk}, \mathbf{r}, \mathbf{R}^0, \mathbf{b})$:

1 : Sample a random tape ρ
2 : $(\widetilde{C}, e, d) \leftarrow \mathsf{G.Gr}(\mathsf{S.Sign}(\mathsf{sk}, \cdot\,; \rho))$
3 : Parse $e = (X_{1,0}, \ldots, X_{n,1})$
4 : $R^1_{i,b} \leftarrow R^0_{i,b} \oplus X_{i,b}$
5 : **Return** $(\mathbf{z}, \widetilde{C}, d)$

$\mathsf{Sim}_2(\mathsf{sk}, \mathbf{r}, \mathbf{R}^0, \mathbf{b})$:

1 : Sample a random tape ρ
2 : $(\widetilde{C}, e, d) \leftarrow \mathsf{G.Gr}(\mathsf{S.Sign}(\mathsf{sk}, \cdot\,; \rho))$
3 : Parse $e = (X_{1,0}, \ldots, X_{n,1})$
4 : $R^1_{i,b_i} \leftarrow R^0_{i,b_i} \oplus X_{i,b_i}$
5 : $R^1_{i,1-b_i} \leftarrow \{0,1\}^\lambda$
6 : **Return** $(\mathbf{z}, \widetilde{C}, d)$

$\mathsf{Sim}_3^{\mathsf{S.Sign}(\mathsf{sk}, \cdot)}(\mathbf{r}, \mathbf{R}^0, \mathbf{b})$:

1 : $\sigma \leftarrow \mathsf{S.Sign}(\mathsf{sk}, b_1 \,||\, \ldots \,||\, b_n)$
2 : $\left((X_{i,b_i})_{i=1}^n, \widetilde{C}, d\right) \leftarrow \mathcal{S}(1^\lambda, \sigma, \mathsf{S.Sign})$
3 : $R^1_{i,b_i} \leftarrow R^0_{i,b_i} \oplus X_{i,b_i}$
4 : $R^1_{i,1-b_i} \leftarrow \{0,1\}^\lambda$
5 : **Return** $(\mathbf{z}, \widetilde{C}, d)$

Fig. 14. Sub-procedures used in hybrids H_3, H_4, H_5. We assume H's output and the garbling labels to have bit-length λ. $\mathcal{S}$ is the garbling simulator from Definition 5.

The proof that each pair of hybrids is computationally indistinguishable is essentially the same as that of Theorem 4.

C Postponed Proofs

C.1 Proof of Theorem 1

In this section, we provide the full proof of Theorem 1 whose intuition was discussed in Section 4. We prove that the AS construction from Figure 8 satisfies witness extractability, computational pre-signature adaptability and unlinkability. We prove each property separately.

Witness extractability. Assume there exists an adversary $\mathcal{A}$ breaking the witness extractability of our AS construction. Our goal is to construct an adversary $\mathcal{B}$ breaking the unforgeability of the FS. We do not show the reduction directly. Instead, we first introduce one game hop in which we replace honest NIZK crs and proof generation with simulated ones in our construction. Thereafter we reduce an adversary against the modified scheme to an adversary breaking the unforgeability of FS.

We start with the original witness extractability game for our AS construction and denote this game H_0, c.f. Figure 15.

$\mathsf{Exp}^{\mathsf{we}}_{\mathsf{AS},\mathcal{R},\mathcal{A}}(1^\lambda)$:

1 : $\mathsf{crs}_{\mathsf{AS}} \leftarrow \mathsf{NIZK.G}(1^\lambda)$
2 : $\mathsf{crs}_{\mathsf{FS}} \leftarrow \mathsf{FS.mskSetup}(1^\lambda)$
3 : $\mathsf{crs} \coloneqq (\mathsf{crs}_{\mathsf{AS}}, \mathsf{crs}_{\mathsf{FS}})$
4 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{FS.mskSetup}(1^\lambda)$
5 : $b \leftarrow 1$, $\mathcal{S}_{\mathsf{AS}} = \emptyset$, $\mathcal{T}_{\mathsf{AS}}[m] = \emptyset$
6 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2}(\mathsf{vk}, \mathsf{crs})$
7 : parse $\sigma^* = (\widehat{\sigma}^*, r^*)$
8 : **for all** $(x, \widetilde{\sigma}) \in \mathcal{T}_{\mathsf{AS}}[m^*]$:
9 : parse $\widetilde{\sigma} = (\mathsf{sk}_p, \pi, \rho)$
10 : if $(x, r^* \oplus \rho) \in \mathcal{R}$ set $b \leftarrow 0$
11 : **Return** 1 if the following conditions hold:
12 : $1 \leftarrow \mathsf{FS.Vfy}(\mathsf{vk}, m^* \| r^*, \widehat{\sigma}^*)$
13 : $m^* \notin \mathcal{S}_{\mathsf{AS}}$
14 : $b = 1$

$\mathcal{O}_1(m)$:

1 : $r \leftarrow \{0,1\}^\ell$
2 : $\widehat{\sigma} \leftarrow \mathsf{FS.Sign}(\mathsf{sk}, m \,||\, r)$
3 : $\mathcal{S}_{\mathsf{AS}} \leftarrow \mathcal{S}_{\mathsf{AS}} \cup \{m\}$
4 : **Return** $\sigma = (\widehat{\sigma}, r)$

$\mathcal{O}_2(m, x)$:

1 : $\rho \leftarrow \{0,1\}^\ell$
2 : $p \coloneqq p_{m,x,\rho}$
3 : $\mathsf{sk}_p \leftarrow \mathsf{FS.KeyGen}(\mathsf{crs}_{\mathsf{FS}}, \mathsf{sk}, p; R)$
4 : $x \coloneqq (\mathsf{vk}, p, \mathsf{sk}_p)$
5 : $\pi \leftarrow \mathsf{NIZK.P}(\mathsf{crs}_{\mathsf{AS}}, x, (\mathsf{sk}, R))$
6 : $\widetilde{\sigma} \coloneqq (\mathsf{sk}_p, \pi, \rho)$
7 : $\mathcal{T}_{\mathsf{AS}}[m] \leftarrow \mathcal{T}_{\mathsf{AS}}[m] \cup \{(x, \widetilde{\sigma})\}$
8 : **Return** $\widetilde{\sigma}$

Fig. 15. Game H_0. Recall that $p_{m,x,\rho}(m', r') : (m = m') \wedge (x, r' \oplus \rho) \in \mathcal{R}$.

In game H_1, instead of sampling $\mathsf{crs}_{\mathsf{AS}}$ through the standard $\mathsf{NIZK.G}$ procedure, we use the zero knowledge simulator Sim to generate a crs_0 and a trapdoor

td. This trapdoor is then used by the simulator Sim in the pre-signing oracle $\mathcal{O}_2$ to generate a simulated proof π. The game can be found in Figure 16 with changes compared to H_0 highlighted in grey. As the only changes are the simulated $\mathsf{crs}_{\mathsf{AS}}$ and proof, distinguishing the two games immediately implies breaking the zero-knowledge property of the NIZK.

$\mathsf{Exp}^{\mathsf{we}}_{\mathsf{AS},\mathcal{R},\mathcal{A}}(1^\lambda)$:

1 : $(\mathsf{crs}_0, \mathsf{td}) \leftarrow \mathsf{Sim}(1^\lambda)$
2 : $\mathsf{crs}_{\mathsf{FS}} \leftarrow \mathsf{FS.mskSetup}(1^\lambda)$
3 : $\mathsf{crs} \coloneqq (\ \mathsf{crs}_0\ , \mathsf{crs}_{\mathsf{FS}})$
4 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{FS.mskSetup}(1^\lambda)$
5 : $b \leftarrow 1$, $\mathcal{S}_{\mathsf{AS}} = \emptyset$, $\mathcal{T}_{\mathsf{AS}}[m] = \emptyset$
6 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2}(\mathsf{vk}, \mathsf{crs})$
7 : parse $\sigma^* = (\widehat{\sigma}^*, r^*)$
8 : **for all** $(x, \widetilde{\sigma}) \in \mathcal{T}_{\mathsf{AS}}[m^*]$:
9 : parse $\widetilde{\sigma} = (\mathsf{sk}_p, \pi, \rho)$
10 : if $(x, r^* \oplus \rho) \in \mathcal{R}$ set $b \leftarrow 0$
11 : **Return** 1 if the following conditions hold:
12 : $1 \leftarrow \mathsf{FS.Vfy}(\mathsf{vk}, m^* \| r^*, \widehat{\sigma}^*)$
13 : $m^* \notin \mathcal{S}_{\mathsf{AS}}$
14 : $b = 1$

$\mathcal{O}_1(m)$:

1 : $r \leftarrow \{0,1\}^\ell$
2 : $\widehat{\sigma} \leftarrow \mathsf{FS.Sign}(\mathsf{sk}, m \,||\, r)$
3 : $\mathcal{S}_{\mathsf{AS}} \leftarrow \mathcal{S}_{\mathsf{AS}} \cup \{m\}$
4 : **Return** $\sigma = (\widehat{\sigma}, r)$

$\mathcal{O}_2(m, x)$:

1 : $\rho \leftarrow \{0,1\}^\ell$
2 : $p \coloneqq p_{m,x,\rho}$
3 : $\mathsf{sk}_p \leftarrow \mathsf{FS.KeyGen}(\mathsf{crs}_{\mathsf{FS}}, \mathsf{sk}, p; R)$
4 : $x \coloneqq (\mathsf{vk}, p, \mathsf{sk}_p)$
5 : $\pi_0 \leftarrow \mathsf{Sim}(\mathsf{td}, x)$
6 : $\widetilde{\sigma} \coloneqq (\mathsf{sk}_p, \ \pi_0\ , \rho)$
7 : $\mathcal{T}_{\mathsf{AS}}[m] \leftarrow \mathcal{T}_{\mathsf{AS}}[m] \cup \{(x, \widetilde{\sigma})\}$
8 : **Return** $\widetilde{\sigma}$

Fig. 16. Game H_1, where Sim is the zero-knowledge simulator guaranteed to exist by the zero-knowledge property of NIZK. The changes compared to H_0 are highlighted in grey.

Now we use an adversary $\mathcal{A}$ against H_1 to construct an adversary $\mathcal{B}$ against the unforgeability of the functional signature scheme FS.

Setup: Upon receiving a verification key vk and a common reference string $\mathsf{crs}_{\mathsf{FS}}$ from the FS unforgeability challenger, the reduction runs the Sim of the NIZK to obtain a simulated crs_0 and a trapdoor td and runs $\mathcal{A}$ on $(\mathsf{vk}, (\mathsf{crs}_0, \mathsf{crs}_{\mathsf{FS}}))$.

Signing queries: When $\mathcal{A}$ makes a signing query on m, the reduction samples $r \leftarrow \{0,1\}^\ell$ and asks its own signing oracle to sing $(m \| r)$. The reduction extends the obtained signature $\widehat{\sigma}$ with r and forwards $(\widehat{\sigma}, r)$ to $\mathcal{A}$. I.e., the reduction behaves as described in $\mathcal{O}_1$ except for using its signing oracle to obtain $\widehat{\sigma}$.

Pre-signing queries: When $\mathcal{A}$ makes a pre-signing query on m, x, the reduction proceeds exactly as described in $\mathcal{O}_2$ of H_1, except for replacing the computation of sk_p with a call to its own functional key generation oracle.

Forgery: Let $(m^*, \sigma^* = (\widehat{\sigma}^*, r^*))$ be the forgery output by $\mathcal{A}$. The reduction $\mathcal{B}$ then outputs $(m^* \| r^*, \widehat{\sigma}^*)$ as its own forgery. We need to show that if

$(m^*, (\widehat{\sigma}^*, r^*))$ is a valid AS forgery, then $(m^* \| r^*, \widehat{\sigma}^*)$ is a valid FS forgery. Let us go through all three checks defining a valid AS forgery:

- $\mathsf{AS.Vfy}(\mathsf{vk}, m^*, (\widehat{\sigma}, r^*)) = 1$: By construction of our AS this means that $\mathsf{FS.Vfy}(\mathsf{vk}, m^* \| r^*, \widehat{\sigma}^*) = 1$.
- $m^* \notin \mathcal{S}_{\mathsf{AS}}$: this means that $\mathcal{A}$ never made a signing query on m^* and hence $\mathcal{B}$ never made a signing query $m^* \| \cdot$ to its signing oracle. Hence $m^* \| r^* \notin \mathcal{S}_{\mathsf{FS}}$).
- $b = 1$: that is, if $\mathcal{A}$ asked for some pre-signatures on m^*, then from none of the answered pre-signatures we can extract a valid witness. We need to show that this implies that none of the predicates queried by $\mathcal{B}$ to the functional key generation oracle is satisfied by the forgery message m^* and randomness r^*. We consider three cases:

  No pre-sign query: If $\mathcal{A}$ made no queries to the pre-sign oracle, $\mathcal{B}$ made no queries to the functional key generation oracle and the statement trivially holds. Assume now $\mathcal{A}$ made at least one query to the pre-sign oracle.

  No m^* pre-sign query: Recall that if $\mathcal{A}$ makes a (m, x) query to its pre-sign oracle, $\mathcal{B}$ simulates such query by querying its functional key generation oracle on the predicate $p_{m,x,\rho}$ for a randomly sampled ρ. By definition of $p_{m,x,\rho}$, we have

  $$p_{m,x,\rho}(m', r') = 1 \Leftrightarrow (m' = m) \wedge (x, r' \oplus \rho) \in \mathcal{R}.$$

  Hence, for a pre-sign query (m, x), where $m \neq m^*$, we know that the respective predicate $p_{m,x,\rho}$ will not be satisfied by (m^*, r^*), independently of what x and ρ are.

  m^* **pre-sign queries:** It remains to discuss pre-sign queries to (m^*, x) for some statement x. For those we have

  $$p_{m^*,x,\rho}(m^*, r^*) = 1 \Leftrightarrow (x, r^* \oplus \rho) \in \mathcal{R}.$$

  Since $b = 1$, we know that $(x, r^* \oplus \rho) \notin \mathcal{R}$. Hence $p_{m^*,x,\rho}(m^*, r^*) = 0$.

To conclude, if $(m^*, \sigma^* = (\widehat{\sigma}^*, r^*))$ is a valid forgery in the H_1 game, then $(m^* \| r^*, \widehat{\sigma}^*)$ is a valid forgery on the FS.

Hence, we showed that the advatage of $\mathcal{B}$ winning the FS unforgeability game is that same as $\mathcal{A}$'s adavantage in winning H_1.

To conclude, we have

$$\mathsf{Adv}^{\mathsf{we,AS}}_{\mathcal{A}} \leq \mathsf{Adv}^{\mathsf{un,FS}}_{\mathcal{B}} + \epsilon_{\mathsf{zk}},$$

where $\mathsf{Adv}^{\mathsf{we,AS}}_{\mathcal{A}}$ denotes the advantage of $\mathcal{A}$ in the witness extractability game for AS, $\mathsf{Adv}^{\mathsf{un}}_{\mathcal{B}}$ the advantage of $\mathcal{B}$ in the unforgeability game for FS and ϵ_{zk} is the negligible distinguishing advantage of the zero-knowledge property for the NIZK.

Computational pre-signature adaptability. We reduce computational pre-signature adaptability of our AS construction w.r.t. a hard relation $\mathcal{R}$ to the adaptive soundness of the NIZK used in the construction.

Before we show the reduction, let us recall that NIZK used in the AS construction is for the relation defined in Eq. 1. We restate it here for easier read:

$$\mathcal{R}_{\mathsf{key}} \;=\; \left\{ ((\mathsf{crs}_{\mathsf{FS}}, \mathsf{vk}, p, \mathsf{sk}_p), (\mathsf{sk}, R)) \;:\; \begin{array}{l} (\mathsf{vk}, \mathsf{sk}) = \mathsf{FS.mskSetup}(1^\lambda; \mathsf{sk}), \\ \mathsf{sk}_p = \mathsf{FS.KeyGen}(\mathsf{crs}_{\mathsf{FS}}, \mathsf{sk}, p; R) \end{array} \right\}.$$

Let us denote $\mathcal{L}_{\mathsf{key}}$ the associated language.

Assume there exists a ppt adversary $\mathcal{A}$ breaking computational pre-signature adaptability. We build a reduction $\mathcal{B}$ breaking soundness of the NIZK as follows:

1. Upon receiving a $\mathsf{crs}_{\mathsf{AS}}$ from the soundness challenger, sample $\mathsf{crs}_{\mathsf{FS}} \leftarrow \mathsf{FS.Setup}$, choose $(x, w) \in \mathcal{R}$ (recall $\mathcal{R}$ is efficiently sampleable) and choose an arbitrary message m. Set $\mathsf{crs} := (\mathsf{crs}_{\mathsf{AS}}, \mathsf{crs}_{\mathsf{FS}})$.
2. Run $\mathcal{A}$ on input $(\mathsf{crs}, m, (x, w))$ to obtain a verification key and pre-signature $(\mathsf{vk}, \widetilde{\sigma})$.
3. Parse $\widetilde{\sigma}$ as $(\mathsf{sk}_p, \pi, \rho)$ and define a predicate $p := p_{m,x,\rho}$ (recall that $p_{m,x,\rho}(m', r') = 1 \Leftrightarrow (m' = m \wedge (x, (r' \oplus \rho) \in \mathcal{R}))$).
4. Output $x_{\mathsf{key}} := (\mathsf{vk}, p, \mathsf{sk}_p)$ and $\pi_{\mathsf{key}} := \pi$.

It is immediate to see that if $\widetilde{\sigma}$ is a valid pre-signature, π_{key} is a valid proof for x_{key}:

$$\begin{aligned} \mathsf{AS.pVfy}(\mathsf{crs}, \mathsf{vk}, m, x, \widetilde{\sigma}) = 1 &\Leftrightarrow \mathsf{NIZK.V}(\mathsf{crs}_{\mathsf{AS}}, (\mathsf{vk}, p, \mathsf{sk}_p), \pi) = 1 \\ &\Leftrightarrow \mathsf{NIZK.V}(\mathsf{crs}_{\mathsf{AS}}, x_{\mathsf{key}}, \pi_{\mathsf{key}}) = 1. \end{aligned}$$

Let $\sigma := \mathsf{AS.Adapt}(\mathsf{vk}, m, \widetilde{\sigma}, w)$. It remains to show the implication:

$$\mathsf{AS.Vfy}(\mathsf{vk}, m, \sigma) = 0 \;\Rightarrow\; x_{\mathsf{key}} \notin \mathcal{L}_{\mathsf{key}}.$$

For contradiction, assume $x_{\mathsf{key}} = (\mathsf{vk}, p, \mathsf{sk}_p) \in \mathcal{L}_{\mathsf{key}}$. Then there exist (sk, R) such that

$$(\mathsf{vk}, \mathsf{sk}) = \mathsf{FS.mskSetup}(1^\lambda, \mathsf{sk}) \wedge \mathsf{sk}_p = \mathsf{FS.KeyGen}(\mathsf{crs}_{\mathsf{FS}}, \mathsf{sk}, p; R).$$

By definition of $\mathsf{AS.Adapt}$, we know that $\sigma = (\widehat{\sigma}, r)$, where

$$r := w \oplus \rho, \qquad \widehat{\sigma} \leftarrow \mathsf{FS.Sign}(\mathsf{sk}_p, m \| r).$$

By perfect correctness of the FS, we know that $\mathsf{FS.Vfy}(\mathsf{vk}, m\|r, \widehat{\sigma}) = 1$ and hence also $\mathsf{AS.Vfy}(\mathsf{vk}, m, \sigma) = 1$ which is a contradiction.

To summarize, we showed that if $\mathcal{A}$ breaks computational pre-signature adaptability with probability $\epsilon_{\mathcal{A}}$, then our reduction $\mathcal{B}$ breaks adaptive soundness of the NIZK with probability $\epsilon_{\mathcal{B}} \geq \epsilon_{\mathcal{A}}$, which concludes the proof.

Unlinkability. Finally, we show that if the functional signature scheme FS satisfies one-time functional privacy, our AS is unlinkable.

Assume there exists a ppt adversary $\mathcal{A}$ breaking the unlinkability of AS. We construct a reduction $\mathcal{B}$ breaking the one-time functional privacy of FS as follows:

Setup: Upon receiving $(\mathsf{vk}, \mathsf{sk})$ and $\mathsf{crs}_{\mathsf{FS}}$, the reduction generates a $\mathsf{crs}_{\mathsf{AS}} \leftarrow \mathsf{NIZK.G}(1^\lambda)$, sets $\mathsf{crs} := (\mathsf{crs}_{\mathsf{AS}}, \mathsf{crs}_{\mathsf{FS}})$ and runs $\mathcal{A}$ on $(\mathsf{crs}, \mathsf{vk})$.

Sign/pre-sign queries: Upon receiving a signing query on message m, the reduction simply computes $\sigma \leftarrow \mathsf{AS.Sign}(\mathsf{sk}, m)$ and returns σ. Analogously, upon receiving a pre-sign query for message m and statement x, the reduction computes $\widetilde{\sigma} \leftarrow \mathsf{AS.pSign}(\mathsf{crs}, \mathsf{sk}, m, x)$ and returns $\widetilde{\sigma}$ to $\mathcal{A}$.

Challenge query: Upon receiving the challenge oracle query (m, x, w) for $(x, w) \in \mathcal{R}$, the reduction defines a message $m_{\mathsf{FS}} = m \| r$ for a random $r \leftarrow \{0,1\}^\ell$. It then defines a predicate $p := p_{m,x,\rho}$ for $\rho = r \oplus w$ (recall, that $p_{m,x,\rho}(m', r') \Leftrightarrow (m' = m \wedge (x, r' \oplus \rho) \in \mathcal{R})$). The reduction submit (m_{FS}, p) to the challenger and gets $\widehat{\sigma}$ as an answer. The reduction then sends $\sigma = (\widehat{\sigma}, r)$ to $\mathcal{A}$.

Guess: Once $\mathcal{A}$ produces a guess b', the reduction $\mathcal{B}$ outputs this values as well.

First we argue that $\mathcal{B}$ perfectly simulates the unlinkability game to $\mathcal{A}$. The setup and simulation of signing/pre-signing queries exactly follows the unlinkability game, hence the simulation is perfect in these phases. Let us focus on the challenge query (m, x, w), which results in $\mathcal{B}$ making the (m_{FS}, p) request as described above. Firstly note that $p(m_{\mathsf{FS}}) = 1$ as

$$p(m_{\mathsf{FS}}) = p_{m,x,\rho}(m, r) = p_{m,x,r\oplus w}(m, r) = (m = m \wedge (x, w) \in \mathcal{R}) = 1.$$

Now if $b = 0$, then $\mathcal{B}$ receives a reply

$$\widehat{\sigma} \leftarrow \mathsf{FS.mskSign}(\mathsf{sk}, m_{\mathsf{FS}}).$$

The message m_{FS} is equal to $m \| r$ for a randomly sampled r. Hence, the distribution of $\sigma = (\widehat{\sigma}, r)$ returned to $\mathcal{A}$ is exactly the same as the reply of the challenge oracle in case $b = 0$ (i.e., $\mathsf{AS.Sign}(\mathsf{sk}, m)$).

If $b = 1$, then $\mathcal{B}$ receives a reply

$$\widehat{\sigma} \leftarrow \mathsf{FS.Sign}(\mathsf{FS.KeyGen}(\mathsf{crs}_{\mathsf{FS}}, \mathsf{sk}, p_1), m_{\mathsf{FS}}).$$

Recall that $p := p_{m,x,\rho}$ for $\rho = r \oplus w$. Since r is sampled uniformaly at random, then also ρ is uniformly distributed. Hence $p_{m,x,\rho}$ corresponds to the predicate that would be generated in $\mathsf{AS.pSign}$. Thus, the distribution of $\sigma = (\widehat{\sigma}, r)$ returned to $\mathcal{A}$ is exactly the same as the reply of the challenge oracle in case $b = 1$ (i.e., $\mathsf{AS.Adapt}(\mathsf{vk}, m, \mathsf{AS.pSign}(\mathsf{crs}_{\mathsf{AS}}, \mathsf{sk}, m, x), w)$).

To summarize, the advantage of $\mathcal{B}$ in the one-time functional privacy game for FS is the same as the advantage of $\mathcal{A}$ in the unlinkability game for AS which concludes the proof.

C.2 Multiple Challenge Witness Encryption

Proof of Lemma 1. Given $\mathcal{A}$ breaking the game in Figure 2, we describe a non uniform family of adversaries $\mathcal{B}_\lambda$, inputs x_λ and messages pair $(m_{0,\lambda}, m_{1,\lambda})$, breaking soundness with comparable advantage. We proceed through a sequence of hybrids H_1 to H_{n+1}. In H_j, once the adversary queries $(x_1, \ldots, x_n)$ and messages $(m_{i,0}, m_{i,1})_{i=1}^n$ the challenger computes

$$c_i = \begin{cases} \mathsf{WE.Enc}(1^\lambda, x_i, m_{i,1}) & \text{If } (i \geq j) \wedge (x_i \notin \mathcal{L}_\lambda) \\ \mathsf{WE.Enc}(1^\lambda, x_i, m_{i,0}) & \text{Otherwise.} \end{cases}$$

Note H_1 and H_{n+1} corresponds respectively to the experiment with challenge bit $b = 1$ and $b = 0$. Hence, there exists a $j = j_\lambda$ such that $\mathcal{A}$ distinguishes H_j from H_{j+1} with probability greater than $\frac{1}{n} \cdot \mathsf{Adv}_{\mathcal{A}}(\lambda)$. We can thus find ρ_λ random tape with maximize the probability of $\mathcal{A}$ distinguishing the two games. In particular

$$\left|\Pr\left[1 \leftarrow \mathcal{A}^{\mathsf{H}_j}(1^\lambda; \rho_\lambda)\right] - \Pr\left[1 \leftarrow \mathcal{A}^{\mathsf{H}_{j+1}}(1^\lambda; \rho_\lambda)\right]\right| \;\geq\; \frac{1}{n} \cdot \mathsf{Adv}_{\mathcal{A}}(\lambda).$$

Note that unless $\mathsf{Adv}_{\mathcal{A}}(\lambda) = 0$ this also implies x_j is a false statement (or else the two hybrids are identical). Next, let $(x_i, m_{i,0}, m_{i,1})_{i=1}^n$ the statements and messages $\mathcal{A}(1^\lambda; \rho_\lambda)$ returns initially, and let $x(\lambda) = x_j$, $m_b(\lambda) = m_{j,b}$. Moreover let $\beta_{\lambda,i}$ be 1 if $x_i \in \mathcal{L}$ or 0 otherwise. We can now provide a description of $\mathcal{B}_\lambda$ in Figure 17 breaking soundness for $x(\lambda)$ and $m_0(\lambda), m_1(\lambda)$. Is immediate to see

$\mathcal{B}_\lambda(c)$

1 : Run $\mathcal{A}(1^\lambda; \rho_\lambda) \to (x_i, m_{i,0}, m_{i,1})_{i=1}^n$
2 : **For** $i \in \{1, \ldots, n\}$ with $i \neq j$:
3 : **If** $i \geq j+1$ and $\beta_{\lambda,i} = 0$: $c_i \leftarrow \mathsf{WE.Enc}(1^\lambda, x_i, m_{i,1})$
4 : **Else**: $c_i \leftarrow \mathsf{WE.Enc}(1^\lambda, x_i, m_{i,0})$
5 : $c_j \leftarrow c$
6 : $b' \leftarrow \mathcal{A}(1^\lambda, c_1, \ldots, c_n; \rho_\lambda)$
7 : **Return** b'

Fig. 17. Non uniform adversary for WE's soundness from $\mathcal{A}$.

hat $\mathcal{B}_\lambda$ correctly simulates H_j if c is the encryption of $m_{i,0}$ and H_{j+1} otherwise. Hence $\mathcal{B}_\lambda$ distinguishes the two games with the same advantage of $\mathcal{A}(1^\lambda; \rho_\lambda)$, and in particular

$$\frac{1}{n} \cdot \mathsf{Adv}_{\mathcal{A}}(\lambda) = \mathsf{Adv}_{\mathcal{B}_\lambda}(\lambda) = \mathsf{negl}(\lambda) \quad \Rightarrow \quad \mathsf{Adv}_{\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda).$$