Friedrich-Alexander-Universität
Erlangen-Nürnberg

Faculty of Engineering - Chair of Applied Cryptograpy
Master Thesis in Mechatronics

Security of Blind Conditional Signatures Revisited

Fabian Jost

First Advisor: Prof. Dr. Dominique Schröder
Second Advisor: Paul Gerhart

Erlangen, 23.09.2024

Abstract

Coin-mixing services allow anonymous blockchain transactions by unlinking the sender and receiver. In a quest to remove trust from the mixing service and improve interoperability and efficiency, Tairi et al. [TMM21] proposed the coin-mixing protocol A2L based on atomic asynchronous locks. Glaeser et al. [GMM+22] revised the A2L protocol and introduced the notion of blind conditional signatures (BCS) as the cryptographic core for coin-mixing services, alongside their security definitions. They further gave an improved construction for a secure BCS protocol called A2L+. In their recent work on adaptor signatures, Gerhart et al. [GSST24] exposed a security gap in the A2L+ protocol that allows breaking unforgeability.

In this work, we first revisit the current security definitions of BCS and pinpoint their weaknesses. Building on this analysis, we introduce the enhanced security property of selective-failure blindness and present a provably secure construction in the game-based setting. Our main contributions in detail are:

- **Security properties:** We review the current security properties of blind conditional signatures according to Glaeser et al. [GMM+22] in detail and highlight gaps based on system assumptions that do not always hold and render the scheme insecure under various settings. We propose the strictly stronger security notion of selective-failure blindness based on the work of Fischlin et al. [FS09] that investigated blindness under aborts. Selective-failure blindness ensures that blindness holds even in case of adversarial aborts during the puzzle solver execution. This is an essential security property for blind signature schemes, as it prevents information leakage that could occur based on aborts.

- **Secure Construction:** As the attack by Gerhart et al. [GSST24] showed, the unforgeability security property of the A2L+ construction by Glaeser et al. [GMM+22] can be broken since the adaptor signature definitions by Aumayr et al. [AEE+21] allow creating a scheme with malleable pre-signatures. We give a revised construction that closes this security gap and protects the sender against collusion between the Hub and Bob. Finally, we prove the security of our revised protocol in the game-based setting.

Contents

1 Introduction

Privacy remains an ongoing challenge for most blockchains due to their inherently transparent nature. While the protocols themselves are trustless, any observer can link transactions to each other and therefore learn where the money flows. To combat this a host of privacy-preserving strategies surfaced, one being coin-mixers.

The basic idea behind coin-mixing is rather simple, namely, unlinking the sender and receiver of a transaction by combining multiple transactions of different users. More pressing is probably the question of why this is beneficial and necessary. During the advent of Bitcoin, it was perceived that blockchain transactions were more or less anonymous. All we know about the sender and receiver are their public addresses, which are large hex numbers. Unfortunately, it is not hard to draw a connection between the public address and the user behind it. Simply buying new bitcoins with a credit card will reveal the real-world identity. In a quest to remove the link between real-world users and blockchain public addresses, coin-mixing services have emerged. These protocols offer the ability to transfer the linked coins to a new yet unlinked address and therefore enable anonymity.

Coin-mixing protocols combine numerous transactions to obfuscate the sender and receiver of two transactions. While there are several techniques to achieve this, many of the modern protocols are built around the concept of synchronization puzzles proposed by Heilman et al. [HAB+17]. Synchronization puzzles are composed of a three-party protocol consisting of the sender Alice, the Hub, who acts as the tumbler, and the receiver Bob. Furthermore, most coin-mixing services run in different phases, consisting of a deposit and withdrawal phase. In the deposit phase, numerous users send a transaction to the Hub with an embedded challenge. Later, during the withdrawal phase, anyone knowing the solution to a challenge can withdraw the corresponding coins to a new address.

To achieve wide-ranging compatibility, remove trust from the Hub, and enhance performance, Tairi et al. [TMM21] proposed the A2L protocol, which is based on the concepts of synchronization puzzles and leverages adaptor signatures according to Aumayr et al. [AEE+21] to embed the challenges inside the transaction messages. Glaeser et al. [GMM+22] later refined the A2L protocol and presented the improved A2L+ protocol. Additionally, they introduced blind conditional signatures (BCS) as the cryptographic core for coin-mixing services. In their recent work on dichotomic signatures, Gerhart et al. [GSST24] showed how an adaptor signature with malleable pre-signatures can be constructed due to flaws in the definitions of adaptor signatures by Aumayr et al. [AEE+21], to break unforgeability of the A2L+ protocol.

In this thesis, we will first explain the functionality of coin-mixing services and synchronization puzzles in detail in Chapter 3. Chapter 4 reviews the security properties

and definitions of blind conditional signatures. We will then restate the attack on the A2L+ protocol, give revised security definitions, and propose a refined, provably secure construction of A2L+ in the game-based setting. In detail, the contributions of this work are summarized below:

Security definitions. We analyze the definitions of blind conditional signatures by Glaeser et al. [GMM+22] in detail and show that blindness in the case of aborts is not adequately handled therein. To combat this, we propose a strictly stronger security definition of selective-failure blindness according to Fischlin et al. [FS09]. Selective-failure blindness ensures that blindness holds even in case of adversarial aborts.

Construction. We revisit the attack on A2L+ by Gerhart et al. [GSST24] and reconstruct the adversary and their adaptor signature scheme with malleable pre-signatures. Furthermore, we state the improved security properties according to Gerhart et al. [GSST24] that we require our adaptor signature scheme to fulfill. With that in place, we give a revised construction of A2L+ that additionally ensures blindness even in case of collusion between the Hub and Bob. Finally, we prove the security of our construction in the game-based setting and therefore show that the A2L+ scheme is secure with respect to the system assumptions.

2 Preliminaries

We denote by $\lambda \in \mathbb{N}$ the security parameter and by $x \leftarrow\!\!\$\ X$ the uniform sampling of x from the set X. $y \leftarrow \mathcal{A}(x; r)$ denotes a probabilistic polynomial time algorithm (PPT) that, on input x and randomness $r \leftarrow\!\!\$\ \{0,1\}^*$, outputs y. The randomness is often omitted and only explicitly mentioned when required. We denote $x := A(y)$ a deterministic polynomial time algorithm (DPT). A function $\mathsf{negl} : \mathbb{N} \to \mathbb{R}$ is negligible in n if for every positive integer $k \in \mathbb{N}$, there exists a threshold $n_0 \in \mathbb{N}$ s.t. for all $n \geq n_0$ it holds that $|\mathsf{negl}(n)| \leq n^{-k}$, meaning it vanishes faster than any polynomial. The following preliminaries are inspired by [AEE+21, GMM+22].

2.1 Hard Relations

We define a relation $\mathsf{Rel} : \mathcal{D}_\mathsf{S} \times \mathcal{D}_\mathsf{W} \to \{0,1\}$ as a mapping, where $\mathcal{D}_\mathsf{S}$ is the space of statements and $\mathcal{D}_\mathsf{W}$ is the space of witnesses. Let (Y, y) be a statement/witness pair, with $Y \in \mathcal{D}_\mathsf{S}$ and $y \in \mathcal{D}_\mathsf{W}$. The relation Rel maps (Y, y) to 1 if and only if y is a valid witness for the statement Y. The language $\mathcal{L}_\mathsf{Rel}$ is denoted as $\mathcal{L}_\mathsf{Rel} := \{Y | \exists y : (Y, y) \in \mathsf{Rel}\}$. The relation is hard if it is computationally infeasible on input a statement Y to compute a valid witness y. Furthermore, verifying the validity of a statement/witness pair and sampling a new pair should be computationally easy. The PPT sampling algorithm is denoted as $\mathsf{GenR}(1^\lambda)$ and outputs a random pair $(Y, y) \in \mathsf{Rel}$.

In this work, we rely on hard relations based on the discrete logarithm problem, which is believed to be hard in polynomial time. We therefore define the corresponding hard relation as Rel_{DL} with respect to the language $\mathcal{L}_\mathsf{DL} := \{Y | \exists y \in \mathbb{Z}_p, Y = g^y\}$, where g is a generator for a group $\mathbb{G}$ of prime order p.

2.2 One-More Discrete Logarithm Problem

The *one-more discrete logarithm problem (OMDL)* extends the classic discrete logarithm problem. In the traditional problem, the goal is to find the exponent x given a generator g and an element h in a finite group $\mathbb{G}$, such that $g^x = h$. In the OMDL problem, the adversary is additionally given $q + 1$ random elements $h_1, ..., h_{q+1} \in \mathbb{G}$ and equipped with a discrete logarithm oracle. The goal is to find $q + 1$ discrete logarithms of $h_1, ..., h_{q+1}$ by only querying the oracle q times. We recall the one-more discrete logarithm assumption according to [BFP21, BNPS03] in Definition 1.

Definition 1 (One-More Discrete Logarithm (OMDL) Assumption [BFP21, BNPS03])**.** *Let $\mathbb{G}$ be a uniformly sampled cyclic group of prime order p and g a random generator of $\mathbb{G}$. The one-more discrete logarithm assumption states that for all $\lambda \in \mathbb{N}$ there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all PPT adversaries $\mathcal{A}$ making at most $q = \mathsf{poly}(\lambda)$ queries to $\mathsf{DL}(\cdot)$, it holds that*

$$\Pr\left[\forall i : x_i = r_i \;\middle|\; \begin{array}{c} r_1, \ldots, r_{q+1} \leftarrow\!\$\, \mathbb{Z}_p; \\ \forall i \in [q+1], h_i \leftarrow g^{r_i} \\ \{x_i\}_{i\in[q+1]} \leftarrow \mathcal{A}^{\mathsf{DL}(\cdot)}(h_1, \ldots, h_{q+1}) \end{array}\right] \leq \mathsf{negl}(\lambda),$$

where the $\mathsf{DL}(\cdot)$ oracle takes as input an element $h \in \mathbb{G}$ and returns x such that $h = g^x$.

2.3 Linear-Only Homomorphic Encryption

Definition 2 (Linear-Only Homomorphic Encryption [TCC 204])**.** *Let $\Pi_E := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ be a* IND-CPA*-secure [GM84] public key encryption scheme with message space $\mathcal{M}$ and the following three algorithms:*

- $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KGen}(1^\lambda)$*. The key generation algorithm is a PPT algorithm that, on input the security parameter λ, outputs an encryption key ek and a decryption key dk.*
- $c \leftarrow \mathsf{Enc}(\mathsf{ek}, m)$*. The encryption algorithm takes as input a public encryption key ek and a plaintext message $m \in \mathcal{M}$ and outputs a ciphertext c.*
- $m \leftarrow \mathsf{Dec}(\mathsf{dk}, c)$*. The decryption algorithm takes as input a private decryption key dk and a ciphertext c and outputs a plaintext message m.*

We say that Π_E is linearly homomorphic if there exists some efficiently computable operation $\circ$ such that for every $m_0, m_1 \in \mathcal{M}$ it holds that $\mathsf{Enc}(\mathsf{ek}, m_0) \circ \mathsf{Enc}(\mathsf{ek}, m_1) = \mathsf{Enc}(\mathsf{ek}, m_0 + m_1)$.

The linear-only encryption (LOE) model introduced by [Gro04] is an idealized model in which the adversary is restricted to performing only linear operations on ciphertexts via oracle access instead of their corresponding algorithms. This restriction simplifies the analysis of the A2L+ scheme by limiting the adversary to linear operations, ensuring that properties such as perfect ciphertext randomizability are preserved, while still capturing essential adversarial behavior.

2.4 Non-Interactive Zero-Knowledge Proof

Let Rel be a hard relation with corresponding language $\mathcal{L}_{\mathsf{Rel}}$ according to Section 2.1. A *non-interactive zero-knowledge proof system (*NIZK*)* for the relation Rel allows a party that we call prover to convince a verifier that a statement Y is true without revealing

any other information about the underlying witness or any other secret data used in the proof. In contrast to an interactive zero-knowledge proof system, a NIZK requires no interaction between the verifier and prover to generate a proof. Looking ahead to our construction, we specifically require the non-interactive property, as we need to prove to another party that a ciphertext encrypts a valid witness to an NP statement under an encryption key ek. We define the language used in the A2L+ NIZK proof system as $\mathcal{L}_{\mathsf{NIZK}} := \{(\mathsf{ek}, Y, c) | \exists y \in \mathbb{Z}_p : g^y = Y \wedge c = \Pi_{\mathsf{E}}.\mathsf{Enc}(\mathsf{ek}, y)\}$. Moreover, in this work, we rely on a randomizable NIZK proof system to ensure that the generated proof can be randomized, such that it verifies a randomized ciphertext also encrypts a valid witness to an NP statement under the same encryption key ek used in the original proof. Belenkiy et al. [BCC+09] first formally defined the notion of a randomizable NIZK and gave a construction based on Groth-Sahai proofs [GS08]. We build upon the advancements presented by Ananth et al. [ADKL19] in their framework for fully homomorphic NIZK proof systems.

Definition 3 (Randomizable Non-Interactive Zero-Knowledge Proof [ADKL19])**.** *Let* $\mathsf{Rel} : \mathcal{D}_S \times \mathcal{D}_W \to \{0,1\}$ *be a hard relation with corresponding language* $\mathcal{L}_{\mathsf{Rel}}$*. A* NIZK *proof system consists of the following four algorithms:*

- $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{Setup}(1^\lambda)$*. The setup algorithm is a* PPT *algorithm that, on input the security parameter* λ*, outputs a common reference string* crs*, used by both the prover and verifier, and a trapdoor* td*.*
- $\pi \leftarrow \mathsf{P}(\mathsf{crs}, Y, y)$*. The prove algorithm is a* PPT *algorithm that takes as input the common reference string* crs*, a statement* Y *that the prover wishes to prove, and a witness* y*. It outputs a proof* π*.*
- $\{0,1\} := \mathsf{V}(\mathsf{crs}, Y, \pi)$*. The verify algorithm is a DPT algorithm that takes as input the common reference string* crs*, a statement* Y*, and a proof* π*. It outputs 1 if the proof is correct and 0 otherwise.*
- $\pi' \leftarrow \mathsf{Rand}(\mathsf{crs}, Y, \pi, r)$*. The randomize algorithm is a* PPT *algorithm that takes as input the common reference string* crs*, a statement* Y*, a proof* π *that the prover wishes to randomize, and a randomization factor* r*. It outputs a randomized proof* π'*.*

A secure randomizable NIZK proof system needs to satisfy the following five properties:

- Perfect Completeness. The verifier will accept the proof if the proof π and the statement Y are valid.
- Soundness. It is computationally infeasible to output a valid proof for a statement $Y \notin \mathcal{L}_{\mathsf{Rel}}$.
- Zero-Knowledge. There exists a simulator $\pi' \leftarrow \mathsf{Sim}(\mathsf{td}, Y)$ that can generate a proof π' without access to the witness y, that is computationally indistinguishable from an honestly generated proof $\pi \leftarrow \mathsf{P}(\mathsf{crs}, Y, y)$.

- Secure. With the knowledge of the trapdoor td, a valid witness y to the statement Y can be efficiently extracted.
- Perfect Randomizability. For any proof π, statement/witness pair (Y, y), and common reference string crs, $\pi' \leftarrow \mathsf{Rand}(\mathsf{crs}, Y, \pi, r)$ and $\pi' \leftarrow \mathsf{P}(\mathsf{crs}, Y', y')$ are indistinguishable, where y' and Y' are the randomized versions of y, Y respectively using the randomization factor r.

The formal security definitions are laid out in the work by De Santis et al. [DMP88] and Ananth et al. [ADKL19].

2.5 Randomizable Puzzle

Randomizable puzzles, first formally defined by Tairi et al. [TMM21], are one of the main pillars that enable anonymous transactions in our coin-mixing protocol. They consist of a cryptographic puzzle, which is a mathematical problem characterized by public parameters, and a cryptographic challenge that must be solved by providing a valid solution to the problem. A randomizable puzzle retains its cryptographic challenge when its parameters are randomized. This allows for generating a computationally indistinguishable new instance from an existing puzzle without changing the underlying challenge. In Definition 4, we restate the definitions and constructions from Tairi et al. [TMM21], used in the A2L+ protocol.

Definition 4 (Randomizable Puzzle [TMM21])**.** *A randomizable puzzle scheme* Π_{RP} *=* $(\mathsf{PSetup}, \mathsf{PGen}, \mathsf{PSolve}, \mathsf{PRand})$ *with a solution space* $\mathcal{S}$ *(and a function* ϕ *action on* $\mathcal{S}$*) consists of four algorithms defined as:*

- $(\mathsf{pp}, \mathsf{td}) \leftarrow \mathsf{PSetup}(1^\lambda)$*. The puzzle setup algorithm is a* PPT *algorithm that on input the security parameter* 1^λ*, outputs public parameters* pp *and a trapdoor* td*.*
- $Z \leftarrow \mathsf{PGen}(\mathsf{pp}, \zeta)$*. The puzzle generate algorithm is a* PPT *algorithm that on input the public parameters* pp *and a puzzle solution* ζ*, outputs a puzzle* Z*.*
- $\zeta := \mathsf{PSolve}(\mathsf{td}, Z)$*. The puzzle solve algorithm is a* DPT *algorithm that on input a trapdoor* td *and a puzzle* Z*, outputs a puzzle solution* ζ*.*
- $(Z', r) \leftarrow \mathsf{PRand}(\mathsf{pp}, Z)$*. The puzzle randomize algorithm is a* PPT *algorithm that on input the public parameters* pp *and a puzzle* Z *(which has a solution* ζ*), outputs a randomization factor* r *and a randomized puzzle* Z' *(which has a solution* $\phi(\zeta, r)$*).*

In their construction, Tairi et al. [TMM21] used a linear-only homomorphic encryption scheme Ψ according to Section 2.3. The construction can be instantiated over any group $\mathbb{G}$, for which the discrete logarithm problem is assumed to be hard. Tairi et al. [TMM21] set the group $\mathbb{G}$ to be an elliptic curve group of order q and

the linear-only homomorphic encryption scheme Ψ to a Castagnos-Laguillaumie (CL) [CL15] encryption scheme with message space $\mathcal{M} = \mathbb{Z}_q$ and a resulting solution space $\mathcal{S} = \mathbb{Z}_q$. We state the proposed construction in Construction 1.

Construction 1 (Randomizable Puzzle [TMM21])**.** *Let $\mathcal{G}$ be a polynomial-time algorithm that takes as input 1^λ and outputs a description of a cyclic group $\mathbb{G}$ of prime order q and a generator g. Let $\Pi_E := (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ be an* IND-CCA*-secure public key encryption scheme.*

- *$\underline{(pp, td) \leftarrow \mathsf{PSetup}(1^\lambda)}$. Run $\mathcal{G}(1^\lambda)$ to obtain $gp := (\mathbb{G}, q, g)$. Sample a key pair $(\mathsf{sk}^\Psi, \mathsf{pk}^\Psi) \leftarrow \mathsf{KGen}(1^\lambda)$, set $pp := (gp, \mathsf{pk}^\Psi)$ and $td := \mathsf{sk}^\Psi$, and return (pp, td).*
- *$\underline{Z \leftarrow \mathsf{PGen}(pp, \zeta)}$. Parse pp as (gp, pk^Ψ), compute $A = g^\zeta$ and $c \leftarrow \Pi_E.\mathsf{Enc}(\mathsf{pk}^\Psi, \zeta)$, and return $Z := (A, c)$.*
- *$\underline{\zeta := \mathsf{PSolve}(td, Z)}$. Parse td as sk^Ψ and Z as (A, c), compute $\zeta \leftarrow \Pi_E.\mathsf{Dec}(\mathsf{sk}^\Psi, c)$, and return ζ.*
- *$\underline{(Z', r) \leftarrow \mathsf{PRand}(pp, Z)}$. Parse Z as (A, c), parse pp as (gp, pk^Ψ), sample $r \leftarrow\!\!\$\ \mathcal{S}$, compute $A' = A \cdot g^r$ and $c' = c \cdot \Pi_E.\mathsf{Enc}(\mathsf{pk}^\Psi, r)$, set $Z' := (A', c')$, and return (Z', r).*

2.6 Commitment Scheme

Definition 5 (Commitment Scheme [Ped92])**.** *A commitment scheme $\Pi_{COM} = (P_{\Pi_{COM}}, V_{\Pi_{COM}})$ consists of the following two algorithms defined as:*

- *$\underline{(com, decom) \leftarrow P_{\Pi_{COM}}(m)}$. The commitment algorithm is a* PPT *algorithm that on input the message m, outputs the commitment com and the decommitment information $decom$.*
- *$\underline{\{0, 1\} := V_{\Pi_{COM}}(com, decom, m)}$. The verification algorithm is a DPT algorithm that on input the commitment com, the decommitment information $decom$, and the message m, outputs 1 if the commitment is valid and 0 otherwise.*

A *commitment scheme* Π_{COM} allows a prover to commit to a message m while keeping its value secret. A verifier can be convinced that the message m was committed by revealing the decommitment information decom and the message m. We call the scheme hiding if the commitment com does not reveal any information about the message m to the verifier. The scheme further ensures that a commitment com to a message m is binding, meaning that once created, the prover cannot change the message. More formally:

Definition 6 (Information-Theoretically Hiding [KL14])**.** *A commitment scheme* Π_{COM} *is information-theoretically hiding if, for any two messages* m_0 *and* m_1 *in the message space* $\mathcal{M}$*, the distributions of the commitments to* m_0 *and* m_1 *are identical. For all adversaries* $\mathcal{A}$ *it holds that*

$$\Pr\left[\mathit{Hiding}^{\mathcal{A}}_{\Pi_{COM}} = 1\right] = 1/2,$$

where the security game $\mathit{Hiding}^{\mathcal{A}}_{\Pi_{COM}}$ *is defined in Figure 2.1, and the probability is taken over the random choices of all probabilistic algorithms.*

Definition 7 (Computationally Binding [KL14])**.** *A commitment scheme* Π_{COM} *is computationally binding if, for all* PPT *adversaries* $\mathcal{A}$*, there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for every* $\lambda \in \mathbb{N}$

$$\Pr\left[\mathit{Binding}^{\mathcal{A}}_{\Pi_{COM}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda),$$

where the security game $\mathit{Binding}^{\mathcal{A}}_{\Pi_{COM}}$ *is defined in Figure 2.1, and the probability is taken over the random choices of all probabilistic algorithms.*

Figure 2.1: Commitment hiding and binding security experiments.

$\mathsf{Hiding}^{\mathcal{A}}_{\Pi_{\mathsf{COM}}}$	$\mathsf{Binding}^{\mathcal{A}}_{\Pi_{\mathsf{COM}}}(\lambda)$
1 : $(m_0, m_1) \leftarrow \mathcal{A}$	1 : $(\mathsf{com}, m, m') \leftarrow \mathcal{A}$
2 : $b \leftarrow \{0,1\}$	2 : $b_0 := (m \neq m')$
3 : $\mathsf{com} \leftarrow \mathsf{P}_{\Pi_{\mathsf{COM}}}(m_b)$	3 : $b_1 := (\Pi_{\mathsf{COM}}(m) = \mathsf{com} = \Pi_{\mathsf{COM}}(m'))$
4 : $b' \leftarrow \mathcal{A}(com)$	4 : **return** $b_0 \wedge b_1$
5 : **return** $(b = b')$	

In this work, we utilize the Pedersen commitment scheme [Ped92], known for its information-theoretically hiding and computationally binding properties. This means that the commitment com perfectly hides the committed message, and even a computationally unbounded adversary cannot determine any information about m from com. However, the scheme is binding only against computationally bounded adversaries, making it infeasible to open the commitment to a different message.

2.7 Digital Signatures

Definition 8 (Digital Signatures [KL14])**.** *A digital signature scheme* Π_{DS} *with message space* $\mathcal{M}$ *is composed of three algorithms* $\Pi_{DS} = (\mathsf{KGen}, \mathsf{Sign}, \mathit{Vrf})$ *defined as:*

- $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$*. The key generation algorithm is a* PPT *algorithm, that on input the security parameter* λ*, outputs a verification key* vk *and a secret key* sk*.*

- $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$. *The signing algorithm is a* PPT *algorithm, that on input a secret key* sk *and message* $m \in \mathcal{M}$*, outputs a signature* σ*.*

- $b := \mathit{Vrf}(\mathsf{vk}, m, \sigma)$. *The verification algorithm is a DPT algorithm that, on input, a verification key* vk*, message* $m \in \mathcal{M}$*, and signature* σ*, outputs a bit* b*.*

It is required that except with negligible probability over $(\mathsf{vk}, \mathsf{sk})$ *output by* $\mathsf{KGen}(1^\lambda)$*, it holds that* $\Pr\Big[\mathit{Vrf}(\mathsf{vk}, \mathsf{Sign}(\mathsf{sk}, m), m) = 1 \,\Big|\, (\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)\Big] = 1$ *for every* $m \in \mathcal{M}$*.*

Of particular interest for us is the security notion of unforgeability. We distinguish between existential (EUF-CMA) and strong (SUF-CMA) unforgeability under adaptive chosen message attacks, which have both been first formally defined by Goldwasser et al. [GMR88]. Roughly speaking, EUF-CMA ensures that a PPT adversary $\mathcal{A}$ cannot create a valid signature σ on a message m that has not been previously signed, even if it obtains signatures on polynomially many messages of its choice other than the challenge message m. More formally, we define existential unforgeability as in Definition 9.

Figure 2.2: Unforgeability security experiments for digital signature schemes.

$\underline{\text{EUF-CMA}^{\mathcal{A}}_{\Pi_{\mathsf{DS}}}(\lambda)}$

1 : $Q := \emptyset$
2 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\Pi_{\mathsf{DS}}}(1^\lambda)$
3 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{vk})$
4 : $b^* = \mathsf{Vrf}(\mathsf{vk}, m^*, \sigma^*) \wedge ((m^*, \cdot) \notin Q)$
5 : **return** b^*

$\underline{\text{SUF-CMA}^{\mathcal{A}}_{\Pi_{\mathsf{DS}}}(\lambda)}$

1 : $Q := \emptyset$
2 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}_{\Pi_{\mathsf{DS}}}(1^\lambda)$
3 : $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{vk})$
4 : $b^* = \mathsf{Vrf}(\mathsf{vk}, m^*, \sigma^*) \wedge ((m^*, \sigma^*) \notin Q)$
5 : **return** b^*

$\underline{\mathcal{O}\mathsf{Sign}(\mathsf{sk}, m)}$

1 : $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$
2 : $Q := Q \cup \{(m, \sigma)\}$
3 : **return** σ

Definition 9 (Existential Unforgeability [GMR88])**.** *A signature scheme* $\Pi_{DS} = (\mathsf{KGen}, \mathsf{Sign}, \mathit{Vrf})$ *is existentially unforgeable under an adaptive chosen-message attack, or just unforgeable if for all* PPT *adversaries* $\mathcal{A}$*, there is a negligible function* $\mathsf{negl}(\lambda)$ *such that for every* $\lambda \in \mathbb{N}$

$$\Pr\Big[\text{EUF-CMA}^{\mathcal{A}}_{\Pi_{DS}}(\lambda) = 1\Big] \leq \mathsf{negl}(\lambda),$$

where the security game $\text{EUF-CMA}^{\mathcal{A}}_{\Pi_{DS}}$ *is defined in Figure 2.2, and the probability is taken over the random choices of all probabilistic algorithms.*

Glaeser et al. [GMM+22] specifically require the digital signature scheme to satisfy the strictly stronger security notion of strong unforgeability under adaptive chosen message attacks (SUF-CMA). SUF-CMA ensures that even if a PPT adversary observes a signature for a specific message m, it cannot produce a different valid signature on that same message m. Strong unforgeability is formally defined as in Definition 10.

Definition 10 (Strong Unforgeability [GMR88])**.** *A signature scheme* $\Pi_{DS} = (\mathsf{KGen}, \mathsf{Sign}, \mathit{Vrf})$ *is strongly unforgeable under an adaptive chosen-message attack, or strongly unforgeable if for all* PPT *adversaries* $\mathcal{A}$*, there is a negligible function* $\mathsf{negl}(\lambda)$ *such that for every* $\lambda \in \mathbb{N}$

$$\Pr\left[\text{SUF-CMA}^{\mathcal{A}}_{\Pi_{DS}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda),$$

where the security game $\text{SUF-CMA}^{\mathcal{A}}_{\Pi_{DS}}$ *is defined in Figure 2.2, and the probability is taken over the random choices of all probabilistic algorithms.*

In the security games for $\text{EUF-CMA}^{\mathcal{A}}_{\Pi_{\mathsf{DS}}}$ and $\text{SUF-CMA}^{\mathcal{A}}_{\Pi_{\mathsf{DS}}}$ in Figure 2.2, the adversary is given access to the signing oracle $\mathcal{O}\mathsf{Sign}$ in both experiments.

In this work, we rely on the Schnorr signature scheme [Sch91], which can be proven secure in the random-oracle model. We recall the construction according to [KL14] in Construction 2.

Construction 2 (Schnorr Signature Scheme [KL14])**.** *Let* $\mathcal{G}$ *be a polynomial-time algorithm that takes as input* 1^λ *and outputs a description of a cyclic group* $\mathbb{G}$ *of prime order* q *and a generator* g*.*

- $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}$*: Run* $\mathcal{G}(1^\lambda)$ *to obtain* $(\mathbb{G}, q, g)$*. Choose a uniform* $\mathsf{sk} \in \mathbb{Z}_q$ *and set* $\mathsf{vk} := g^{\mathsf{sk}}$*. The secret key is* sk*, and the verification key is* $(\mathbb{G}, q, g, \mathsf{vk})$*. As part of key generation, a function* $H : \{0,1\}^* \to \mathbb{Z}_q$ *is specified.*
- $(R, s) \leftarrow \mathsf{Sign}$*: On input a secret key* sk *and a message* $m \in \{0,1\}^*$*, choose uniform* $k \in \mathbb{Z}_q$ *and set* $R := g^k$*. Then compute* $e := H(R, \mathsf{vk}, m)$*, followed by* $s := [e \cdot \mathsf{sk} + k \bmod q]$*. Output the signature* (R, s)*.*
- $\{0,1\} := \mathit{Vrf}$*: On input a verification key* $(\mathbb{G}, q, g, \mathsf{vk})$*, a message* m *and a signature* (R, s)*, compute* $e := H(R, \mathsf{vk}, m)$*. Output 1 if* $g^s = R \cdot \mathsf{vk}^e$ *and 0 otherwise.*

2.8 Adaptor Signatures

In most standard digital signature schemes, the signer directly commits to a given message. This is perfectly feasible and desirable in general-purpose applications like signing a document but leaves little to no opportunities for more advanced use cases. If, for example, a signer only wants to commit to a message or transaction given that certain criteria or circumstances are met, there is no possibility of achieving that with standard digital signature schemes alone.

To circumvent that and enable a host of other applications, *adaptor signatures* have first been proposed by Poelstra [Poe17] and later formally defined by Aumayr et al. [AEE+21] in a payment channel setting. Initially, they were designed for a one-time fair exchange of a coin given a witness. Because of their potential in other areas, they are now being used in various other settings. This inherently begged the question of whether the security model is strong enough for these diverging use cases.

Dai et al. [DOY22] consequently discovered flaws in the aforementioned definitions. Interestingly, these faulty definitions have still been in use until a revised construction was proposed by Gerhart et al. [GSST24], based on their novel dichotomic signature scheme abstraction.

The security gaps discovered are lethal for blind conditional signatures and the A2L+ coin-mixing protocol proposed by Glaeser et al. [GMM+22]. We therefore revisit the original definitions by Aumayr et al. [AEE+21] and then progress towards the provably secure definitions used for dichotomic adaptor signatures by Gerhart et al. [GSST24].

2.8.1 Definition and Functionality

Adaptor signature schemes were defined in [AEE+21, GSST24] with respect to a digital signature scheme Π_{DS} and a hard relation Rel.

They allow the signer holding a secret key sk to create a pre-signature $\tilde{\sigma}$ on any message m w.r.t. a statement $Y \in \mathcal{L}_{\mathsf{Rel}}$. The pre-signature algorithm pSign therefore encrypts a witness y from the statement Y in the pre-signature $\tilde{\sigma}$. The pVrf algorithm allows verifying that $\tilde{\sigma}$ can be adapted to a valid full signature σ and that σ is a valid signature for m. Given the witness y, $\tilde{\sigma}$ can be adapted to a full signature σ on the message m using the Adapt algorithm. Additionally, the witness can be extracted with the $\mathsf{Extract}$ algorithm, given σ and $\tilde{\sigma}$.

According to the definition of Gerhart et al. in [GSST24], the scheme as proposed in [AEE+21] consists of the four algorithms stated in Definition 11.

Definition 11 (Adaptor Signature)**.** *An adaptor scheme Π_{AS} w.r.t. a hard relation Rel and a signature scheme $\Pi_{DS} = (\mathsf{KGen}, \mathsf{Sign}, Vrf)$ consists of four algorithms $\Pi_{AS} = (pSign, Adapt, pVrf, Extract)$ defined as:*

- *$\tilde{\sigma} \leftarrow pSign(\mathsf{sk}, m, Y)$. The pre-signing algorithm is a PPT algorithm that on input a secret key sk, message $m \in \{0,1\}^{l_m}$ and statement $Y \in \mathcal{L}_{Rel}$, outputs a pre-signature $\tilde{\sigma}$.*
- *$b := pVrf(\mathsf{vk}, m, Y, \tilde{\sigma})$. The pre-verification algorithm is a DPT algorithm that on input a verification key vk, message $m \in \{0,1\}^{l_m}$, statement $Y \in \mathcal{L}_{Rel}$, and pre-signature $\tilde{\sigma}$, outputs a bit b.*
- *$\sigma := Adapt(\mathsf{vk}, \tilde{\sigma}, y)$. The adapting algorithm is a DPT algorithm that on input a verification key vk, pre-signature $\tilde{\sigma}$, and witness y for the statement $Y \in \mathcal{L}_{Rel}$, outputs an adapted signature σ.*

- $y := \mathsf{Extract}(\mathsf{vk}, \tilde{\sigma}, \sigma, Y)$. *The extracting algorithm is a* DPT *algorithm that on input a verification key* vk*, pre-signature* $\tilde{\sigma}$*, signature* σ*, and statement* $Y \in \mathcal{L}_{\mathsf{Rel}}$*, outputs a witness* y *such that* $(Y, y) \in \mathsf{Rel}$*, or* $\perp$*.*

Figure 2.3: Adaptor signatures for one-time fair coin exchange.

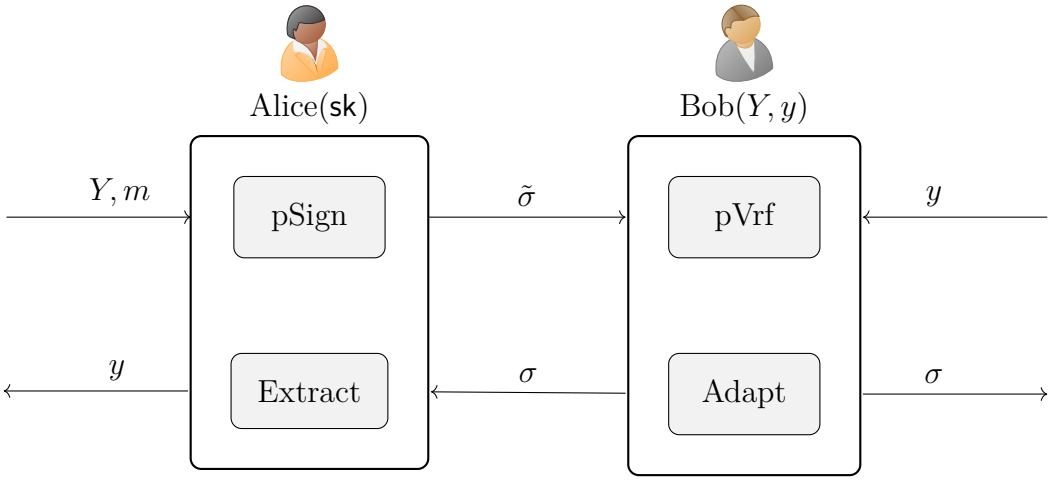


Figure 2.3 illustrates the functionality of adaptor signatures for a fair coin exchange. In this scenario, Alice wants to learn a secret to an NP statement Y held by Bob in exchange for a signature. She uses the pSign algorithm to pre-sign a transaction message m w.r.t. Y. Bob then first computes the pVrf algorithm to verify the correctness of the pre-signature $\tilde{\sigma}$. Given that it is valid, Bob uses his secret y to adapt the pre-signature $\tilde{\sigma}$ to a full signature σ with the help of the Adapt algorithm. Afterward, he publishes the full signature σ and thereby receives his payment. Given the full signature σ and the pre-signature $\tilde{\sigma}$, Alice can now leverage the $\mathsf{Extract}$ algorithm and learn the secret value y.

2.8.2 Security Properties of Adaptor Signatures

As Gerhart et al. [GSST24] pointed out, the security definitions introduced by Aumayr et al. [AEE+21] are not chosen strong enough for use cases outside a payment channel setting. For us, especially the witness extractability is not chosen strong enough and allows the adversary to generate malleable and leaky pre-signatures. We will see later why this is an issue and show why, in our revised definition of blind conditional signatures, this cannot be allowed. Dai et al. [DOY22] introduced the stronger security notions of extractability and unique extractability, which we restate in this section alongside the other security properties, following the work of Gerhart et al. [GSST24].

Pre-Signature Correctness

Intuitively, pre-signature correctness ensures that any honestly generated pre-signature $\tilde{\sigma}$ on a message m under the secret key sk with an embedded honest statement

$(Y, y) \in \mathsf{Rel}$ and any honestly adapted signature σ can be successfully verified with the corresponding public verification key vk. More formally:

Definition 12 (Pre-Signature Correctness)**.** *An adaptor signature Π_{AS} satisfies pre-signature correctness, if for all $\lambda \in \mathbb{N}$, every $m \in \{0,1\}^{l_m}$ and every statement/witness pair $(Y, y) \in \mathsf{Rel}$, it holds that*

$$\Pr\left[\begin{array}{c|c} \mathsf{pVrf}(\mathsf{vk}, m, Y, \tilde{\sigma}) = 1 & (\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda), \\ \wedge & (Y, y) \leftarrow \mathsf{GenR}(1^\lambda)(1^\lambda), \\ \mathsf{Vrf}(\mathsf{vk}, m, \sigma) = 1 & \tilde{\sigma} \leftarrow \mathsf{pSign}(\mathsf{sk}, m, Y), \\ \wedge & \sigma := \mathsf{Adapt}(\mathsf{vk}, \tilde{\sigma}, y), \\ (Y, y') \in \mathsf{Rel} & y' := \mathsf{Extract}(\mathsf{vk}, \tilde{\sigma}, \sigma, Y) \end{array}\right] = 1.$$

Pre-Signature Adaptability

Intuitively, pre-signature adaptability guarantees that any verifiable pre-signature $\tilde{\sigma}$ on the message m under the verification key vk can be adapted to a verifiable full signature σ on the message m by knowing the witness y to the embedded honest statement Y. More formally:

Definition 13 (Pre-Signature Adaptability)**.** *An adaptor signature scheme Π_{AS} satisfies pre-signature adaptability, if for all $\lambda \in \mathbb{N}$, messages $m \in \{0,1\}^*$, statement/witness pairs $(Y, y) \in \mathsf{Rel}$, public keys vk and pre-signatures $\tilde{\sigma} \in \{0,1\}^*$ we have $\mathsf{pVrf}(\mathsf{vk}, m, Y, \tilde{\sigma}) = 1$, then $\mathsf{Vrf}(\mathsf{vk}, m, \mathsf{Adapt}(\mathsf{vk}, \tilde{\sigma}, y)) = 1$.*

Extractability

Intuitively, extractability ensures that given a valid adapted signature σ and corresponding pre-signature $\tilde{\sigma}$ for the same message m, a valid witness y to the embedded honest statement $Y \in \mathsf{Rel}$ can be extracted. Dai et al. [DOY22] extended this notion to the multiple query setting, where the adversary is allowed to see multiple pre-signatures and pre-signatures on multiple honestly sampled statements. Extractability inherently implies the unforgeability of adaptor signatures. Consequently, in order to demonstrate that an adversary $\mathcal{A}$ is capable of producing a valid forgery within the security proofs in Chapter 8, we establish that $\mathcal{A}$ can successfully break the extractability property. The adversary wins the $\mathsf{Extractability}^{\mathcal{A}}_{\Pi_{\mathsf{AS}}}$ game by outputting a special forgery (m^*, σ^*) that does not allow successfully extracting a witness y. More formally:

Definition 14 (Extractability)**.** *An adaptor signature scheme Π_{AS} is extractable, if for every PPT adversary $\mathcal{A}$ there exists a negligible function $\mathsf{negl}(\lambda)$ such that for every $\lambda \in \mathbb{N}$*

$$\Pr\left[\mathsf{Extractability}^{\mathcal{A}}_{\Pi_{\mathsf{AS}}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda),$$

where experiment $\mathsf{Extractability}^{\mathcal{A}}_{\Pi_{\mathsf{AS}}}$ is described in Figure 2.4, and the probability is taken over the random choices of all probabilistic algorithms.

Figure 2.4: $\mathsf{Extractability}^{\mathcal{A}}_{\Pi_{\mathsf{AS}}}(\lambda)$ security game.

$\mathsf{Extractability}^{\mathcal{A}}_{\Pi_{\mathsf{AS}}}(\lambda)$

$(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$b \leftarrow 1$
$\mathcal{S}, \mathcal{C}, \mathcal{T} \leftarrow \emptyset$
$(m^*, \sigma^*) \leftarrow \mathcal{A}(\mathsf{vk})^{\mathsf{NewY}(1^\lambda), \mathsf{pSign}(\mathsf{sk},\cdot), \mathsf{Sign}(\mathsf{sk},\cdot)}$
assert $\mathsf{Vrf}(\mathsf{vk}, m^*, \sigma^*)$
assert $(m^* \notin \mathcal{S})$
for $(Y, \widetilde{\sigma}) \in \mathcal{T}[m^*]$
  if $(Y, \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma^*, Y)) \in \mathsf{Rel}$ **then**
    $b \leftarrow 0$
return b

$\mathsf{pSign}(\mathsf{sk}, m, Y)$

$\widetilde{\sigma} \leftarrow \mathsf{pSign}(\mathsf{sk}, m, Y)$
$\mathcal{T}[m] \leftarrow \mathcal{T}[m] \cup \{(Y, \widetilde{\sigma})\}$
return $\widetilde{\sigma}$

$\mathsf{Sign}(\mathsf{sk}, m)$

$\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$
$\mathcal{S} \leftarrow \mathcal{S} \cup \{m\}$
return σ

$\mathsf{NewY}(\lambda)$

$(Y, y) \leftarrow \mathsf{Rel.GenR}(1^\lambda)$
$\mathcal{C} \leftarrow \mathcal{C} \cup \{Y\}$
return Y

Unique Extractability

Intuitively, unique extractability prevents malleable pre-signatures, where two valid signatures can be generated from the same pre-signature. In Section 5.2, we will show how Gerhart et al. [GSST24] constructed an adversary that is able to create malleable pre-signatures but is secure according to the definitions of Aumayr et al. [AEE+21]. Roughly speaking, unique extractability states that any verifying pre-signature creates a commitment to a single valid signature. More formally:

Figure 2.5: $\mathsf{UniqueExtractability}^{\mathcal{A}}_{\Pi_{\mathsf{AS}}}(\lambda)$ security game.

$\mathsf{UniqueExtractability}^{\mathcal{A}}_{\Pi_{\mathsf{AS}}}(\lambda)$

$(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
$(m, Y, \widetilde{\sigma}, \sigma, \sigma') \leftarrow \mathcal{A}^{\mathsf{pSign}(\mathsf{sk},\cdot,\cdot), \mathsf{Sign}(\mathsf{sk},\cdot)}(\mathsf{vk})$
assert $(\sigma \neq \sigma') \wedge \mathsf{Vrf}(\mathsf{vk}, m, \sigma) \wedge \mathsf{Vrf}(\mathsf{vk}, m, \sigma')$
assert $\mathsf{pVrf}(\mathsf{vk}, m, Y, \widetilde{\sigma})$
$y \leftarrow \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma, Y); y' \leftarrow \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}, \sigma', Y)$
return $(Y, y) \in \mathsf{Rel} \wedge (Y, y') \in \mathsf{Rel}$

$\mathsf{pSign}(\mathsf{sk}, m, Y)$

$\widetilde{\sigma} \leftarrow \mathsf{pSign}(\mathsf{sk}, m, Y)$
return $\widetilde{\sigma}$

$\mathsf{Sign}(\mathsf{sk}, m)$

$\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$
return σ

Definition 15 (Unique Extractability)**.** *An adaptor signature scheme* Π_{AS} *is unique extractable, if for every* PPT *adversary* $\mathcal{A}$ *there exists a negligible function* $\mathsf{negl}(\lambda)$ *such*

that for every $\lambda \in \mathbb{N}$

$$\Pr\Big[\mathit{UniqueExtractability}^{\mathcal{A}}_{\Pi_{AS}}(\lambda) = 1\Big] \leq \mathsf{negl}(\lambda),$$

where experiment $\mathit{UniqueExtractability}^{\mathcal{A}}_{\Pi_{AS}}$ *is described in Figure 2.5, and the probability is taken over the random choices of all probabilistic algorithms.*

Unlinkability

Intuitively, unlinkability ensures that no adversary can distinguish an adapted pre-signature from a standard signature. This holds even when the adversary generates the witnesses. More formally:

Figure 2.6: $\mathsf{Unlinkability}^{\mathcal{A}}_{\Pi_{\mathsf{AS}}}(\lambda)$ security game.

$\mathsf{Unlinkability}^{\mathcal{A}}_{\Pi_{\mathsf{AS}}}(\lambda, b)$

1 : $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$
2 : $b' \leftarrow \mathcal{A}^{\mathsf{Chall}(b,\mathsf{sk},\cdot,\cdot),\mathsf{Sign}(\mathsf{sk},\cdot),\mathsf{pSign}(\mathsf{sk},\cdot,\cdot)}(\mathsf{vk})$
3 : **return** b'

$\mathsf{Chall}(b, \mathsf{sk}, m, (Y, y))$

1 : **assert** $(Y.y) \in \mathsf{Rel}$
2 : $\widetilde{\sigma} \leftarrow \Pi_{\mathsf{AS}}.\mathsf{pSign}(\mathsf{sk}, m, Y)$
3 : $\sigma_0 := \mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}, y)$
4 : $\sigma_1 \leftarrow \Pi_{\mathsf{DS}}.\mathsf{Sign}(\mathsf{sk}, m)$
5 : **return** σ_b

$\mathsf{pSign}(\mathsf{sk}, m, Y)$

1 : $\widetilde{\sigma} \leftarrow \Pi_{\mathsf{AS}}.\mathsf{pSign}(\mathsf{sk}, m, Y)$
2 : **return** $\widetilde{\sigma}$

$\mathsf{Sign}(\mathsf{sk}, m)$

1 : $\sigma \leftarrow \Pi_{\mathsf{DS}}.\mathsf{Sign}(\mathsf{sk}, m)$
2 : **return** σ

Definition 16 (Unlinkability)**.** *An adaptor signature scheme* Π_{AS} *is unlinkable, if for every* PPT *adversary* $\mathcal{A}$ *there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for every* $\lambda \in \mathbb{N}$

$$\Big|\Pr\Big[\mathit{Unlinkability}^{\mathcal{A}}_{\Pi_{AS}}(\lambda, 0) = 1\Big] - \Pr\Big[\mathit{Unlinkability}^{\mathcal{A}}_{\Pi_{AS}}(\lambda, 1) = 1\Big]\Big| \leq \mathsf{negl}(\lambda),$$

where experiment $\mathit{Unlinkability}^{\mathcal{A}}_{\Pi_{AS}}$ *is described in Figure 2.6, and the probability is taken over the random choices of all probabilistic algorithms.*

Pre-Verify Soundness

Intuitively, pre-verify soundness ensures that the pre-verification algorithm only evaluates to true if the statement is in the relation and rejects pre-signatures computed using statements $Y \notin \mathsf{Rel}$. Pre-verify soundness complements pre-signature adaptability, which is restricted to honestly generated pre-signatures on statements in the relation $Y \in \mathsf{Rel}$. In Definition 17 and Definition 18, we restate computational and statistical soundness according to Gerhart et al. [GSST24]. More formally:

Definition 17 (Computational Pre-Verify Soundness)**.** *An adaptor signature scheme* Π_{AS} *satisfies computational pre-verify soundness, if for every* PPT *adversary* $\mathcal{A}$ *there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for every* $\lambda \in \mathbb{N}$ *and polynomially bounded* $Y \notin \mathcal{L}_{\mathsf{Rel}}$*,*

$$\Pr\Big[(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda), (m, \widetilde{\sigma}) \leftarrow \mathcal{A}(\mathsf{sk}) : \mathsf{pVrf}(\mathsf{vk}, m, \widetilde{\sigma}, Y) = 1\Big] \leq \mathsf{negl}(\lambda).$$

Definition 18 (Statistical Pre-Verify Soundness)**.** *An adaptor signature scheme* Π_{AS} *satisfies statistical pre-verify soundness, if for every* $\lambda \in \mathbb{N}$*, polynomially bounded* $Y \notin \mathcal{L}_{\mathsf{Rel}}$*, key pair* $\mathsf{vk}, \mathsf{sk})$ *in the support of* $\mathsf{KGen}(1^\lambda)$*,*

$$\Pr[\exists(m, \widetilde{\sigma}) : \mathsf{pVrf}(\mathsf{vk}, m, Y) = 1] = 0.$$

3 Coin-Mixing

As we have already established in the introduction, coin-mixing services allow anonymous transactions of cryptocurrencies by combining multiple transactions of different users. All the proposed protocols rely on a minimum number of users for this to work, as linking the transactions becomes trivial otherwise. Furthermore, coin-mixers do not directly solve the privacy issue on blockchains; they rather offer users the possibility of sending coins to another address securely and privately if they choose to do so. While the idea behind most of the coin-mixing services is identical, the system assumptions and functionality in detail can vary significantly. In this chapter, we will first explore the general idea and functionality behind coin-mixing protocols and then progress toward specific definitions of the A2L+ protocol.

3.1 General Functionality

In the early days there have been numerous coin-mixing protocols like CoinJoin [DS21] or CoinShuffle [RMK14], that allowed a set of mutually distrusting parties to achieve unlinkability by combining their transaction inputs and outputs. While this works perfectly fine in theory, the obvious challenge is finding and connecting enough participants to initiate the protocol.

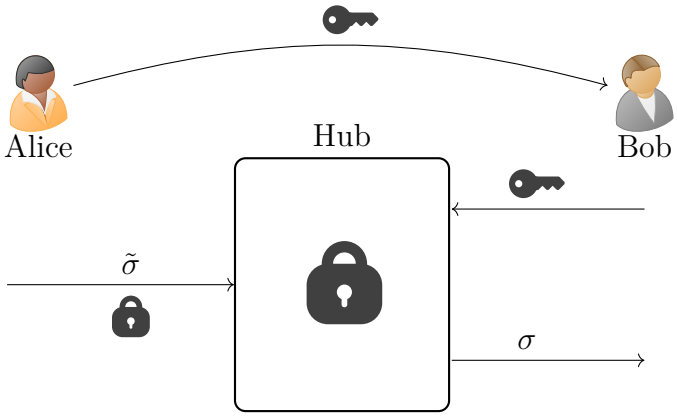
To solve this, third-party coin-mixing services arose. These either connect users to execute the aforementioned protocols or directly act as tumblers themselves. In the latter, users simply send their coins to the third-party hub. Once all coins have been received, the hub sends them back in a randomized order. It is easy to see that in this scenario, the hub can simply go offline and stop sending out the mixed coins, while still accepting incoming requests.

In the pursuit of removing trust from the coin-mixing services, novel protocols have been proposed. Most of these protocols act in essence as privacy-preserving payment channel hubs. In contrast to the previously mentioned protocols and standard payment channel hubs, the coins are not automatically sent to the receiving party but need to be retrieved manually. Additionally, not even the hub should be able to connect the depositor and withdrawer.

Figure 3.1 represents an overly simplified version of the basic functionality of modern coin-mixers. Alice sends coins to the hub alongside a challenge. Once the hub receives the coins, Alice can send the secret to Bob. Bob can then use the secret to solve the challenge and retrieve the locked coins.

In the above-mentioned scheme from Figure 3.1 it is not easy to achieve unlinkability, since the challenge the depositor provided and the one the withdrawer is solving are

Figure 3.1: Coin-mixing simplified.



the same. Some protocols like Tornadocash [PSS19] try to solve this by leveraging zero-knowledge proof systems. These advanced zk-proofs are unfortunately not available on all major blockchains. Since we are aiming at a generic solution, we will introduce a new cryptographic primitive that enables us to overcome these issues and achieve wide-ranging compatibility in the next section.

3.2 Synchronization Puzzles

In their work on Tumblebit, Heilman et al. [HAB+17] proposed a new primitive named *synchronization puzzles* that paved the way towards widely compatible coin-mixing services. The proposed protocol still relies on hashed time-lock contracts (HTLCs), which are incompatible with some major blockchains. We will later see, how Tairi et al. [TMM21] adapted the primitive in their A2L protocol to overcome that. Most notably, synchronization puzzles serve as the basis for blind conditional signatures introduced by Glaeser et al. [GMM+22], which will be analyzed in detail in Chapter 4. In this section, we will focus on the cryptographic primitive's groundwork and examine how it is used in coin-mixing protocols. The following explanation is merely an abstraction of the originally proposed version by [HAB+17], which is tailored to extract and visualize the basic principles.

A synchronization puzzle is a protocol between three parties, referred to here as Alice, Bob, and the Hub. It consists of the two phases *puzzle promise* and *puzzle solver*, which are protocols themselves. The two phases can be abstracted in the following way:

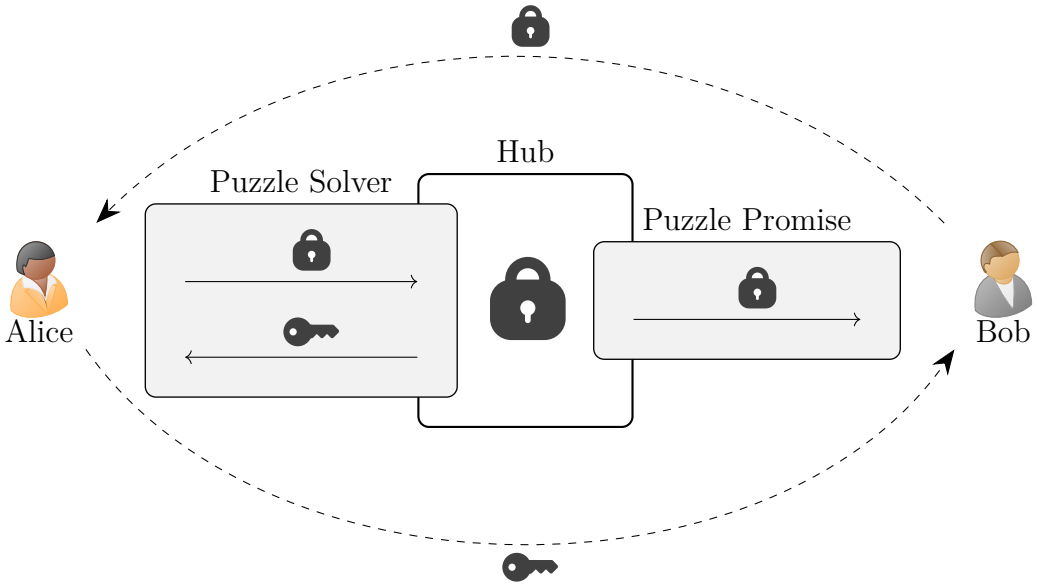
- $(\perp, \{\tau, \perp\}) \leftarrow \mathsf{PPromise}\left\langle \begin{matrix} H\left(\mathsf{sk}^H, m_{HB}\right) \\ B\left(\mathsf{vk}^H, m_{HB}\right) \end{matrix} \right\rangle$: *The puzzle promise protocol is an interactive protocol run between two parties H (with inputs the signing key sk^H, and a message m_{HB}) and B (with inputs the verification key vk^H, and a message m_{HB}). On a request from B, the protocol returns $\perp$ to H and either $\perp$ or a*

puzzle τ with an embedded signature σ on the message m_{HB} under the signing key sk^H to B.

- $(\{(\sigma^*, s), \perp\}, \{\sigma^*, \perp\}) \leftarrow \mathsf{PSolver}\left\langle \begin{matrix} A\left(\mathsf{sk}^A, m_{AH}, \tau\right) \\ H\left(\mathsf{vk}^A, m_{AH}\right) \end{matrix} \right\rangle$: *The puzzle solver protocol is an interactive protocol between two parties A (with inputs the signing key sk^A, a message m_{AH}, and a puzzle τ) and H (with inputs the verification key vk^A, and a message m_{AH}) and returns to both users either a signature σ^* for the message m_{AH} under the signing key sk^A (A additionally receives a secret s) or $\perp$.*

In the beginning, Bob initiates the puzzle promise phase by requesting a puzzle from the Hub with an embedded signature σ for some message m_{HB}. He then sends the puzzle privately to Alice. Alice now initiates the puzzle solver phase, by requesting the solution s to the puzzle from the Hub. In exchange for the puzzle solution s, the Hub receives a signature σ' from Alice with respect to some message m_{AH}. Lastly, Alice sends the solution s to Bob, which allows him to extract the embedded signature σ from the original puzzle. Figure 3.2 illustrates the overall protocol.

Figure 3.2: Synchronization puzzle. Dashed arrows represent secure, private communication.



As Glaeser et al. [GMM+22] intuitively pointed out, such a protocol needs to satisfy the following properties:

- Blindness. The puzzle solver protocol does not leak any information about the puzzle, that could link Alice and Bob.
- Unlockability. The output σ of the puzzle solver protocol that Alice receives from the Hub, enables Bob to solve the original puzzle and retrieve a valid signature to

the original puzzle. The Hub can therefore not unlock the signature σ' without enabling Bob to retrieve the signature σ.

- Unforgeability. A solution to a puzzle can only be obtained by initiating the puzzle solver protocol, meaning Bob can not derive n valid signatures without at least n puzzle solver protocols being completed by the Hub.

3.3 A2L+

With the basic functionalities and underlying primitives laid out, we have everything to put together the A2L+ protocol. Tairi et al. [TMM21] adapted synchronization puzzles mainly in two ways. Firstly, by exchanging hashed time-lock contracts (HTLCs) with adaptor signatures and randomizing the puzzles, and secondly, by adding a registration protocol to protect against griefing attacks. For our analysis of blind conditional signatures, the griefing protection is irrelevant. For coin-mixing services, on the other hand, it is indispensable. Because of this, we will also dive into how the mechanism works but explain it separately from the rest of the protocol.

3.3.1 System Assumptions

To make anonymous payments possible, A2L+ makes a few system assumptions. Some of these are inherited by the underlying blockchain, and others must be considered during implementation. From the underlying blockchain, we require that time locks be available. This is necessary since it needs to be ensured that the coins used as transaction inputs of the messages that are pre-signed in the puzzle promise phase do not get spent in another transaction before the puzzle solver protocol execution finishes. Hence, this ensures that Bob can ultimately claim his coins by publishing the transaction. In A2L+, this is done implicitly by requiring the sender and receiver to set up a payment channel with the Hub in advance. Furthermore, it is assumed that all parties have carried out the key generation procedure beforehand.

The anonymity set is defined by the number of successful protocol executions. We therefore require a minimum number of participants to enable anonymity in the first place. Moreover, a constant amount of coins per transaction is required since linking the sender and receiver otherwise becomes trivial.

Similar to Tumblebit [HAB+17], the A2L+ protocol runs in phases and epochs, where each epoch consists of a registration phase, puzzle promise phase, and puzzle solver phase, respectively. The duration for each epoch and phase can be chosen by the protocol designer. We finally assume that no party colludes with the Hub and that the communication between the sender and receiver is secure, private, and unnoticed by the Hub.

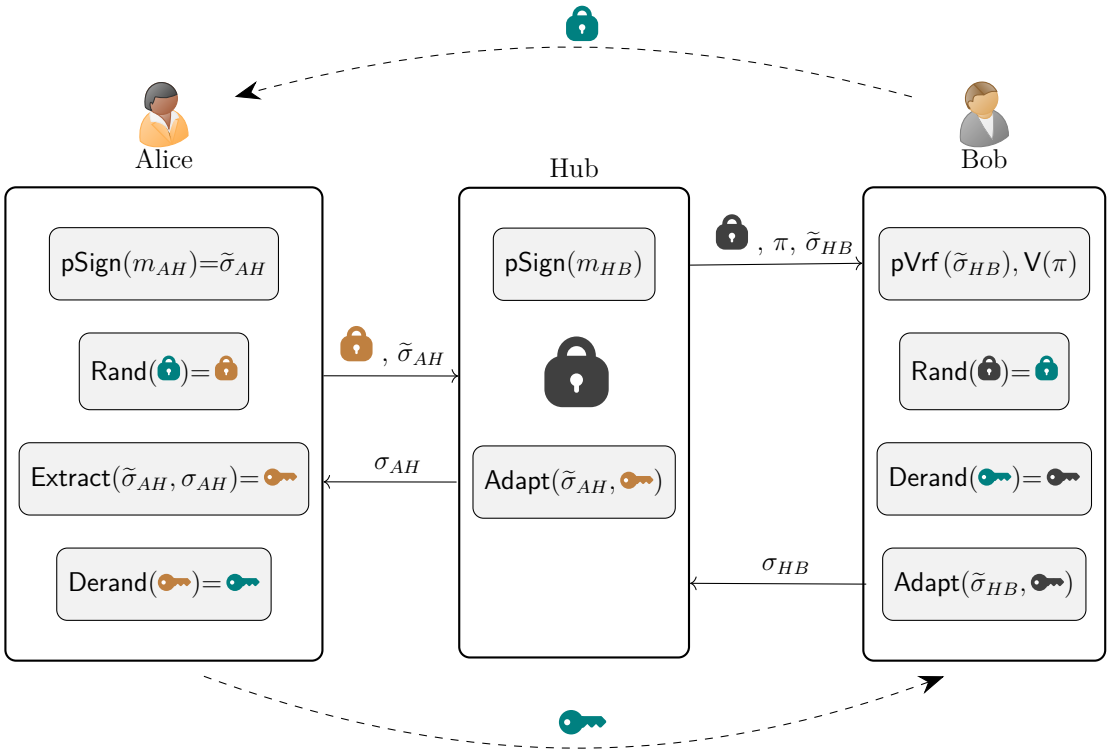
3.3.2 Achieving Unlinkability

Tairi et al. [TMM21] first needed to find a suitable puzzle promise and solver protocol to achieve wide-ranging compatibility. Adaptor signatures defined in Section 2.8 are compatible with most major blockchains, as they only require Schnorr signatures to be available as an underlying cryptographic primitive. They can also be used to simulate the puzzle promise and solver phases in the A2L+ protocol.

One obvious issue that adaptor signatures could not solve is the unlinkability between the puzzle promise and puzzle solver protocol. If Alice and Bob use the same condition for both phases inside the respective adaptor signatures, the Hub can easily link them. To overcome this, Tairi et al. [TMM21] proposed to leverage randomizable puzzles described in Section 2.5 and randomize the embedded condition. This obviously led to the issue that the Hub can now not solve the puzzle in the puzzle solver phase because he does not know the witness to the randomized puzzle. The solution to this is that the Hub needs to extend the puzzle with the encryption of the witness under his secret key, using a linear-only homomorphic encryption scheme described in Section 2.3. When the Hub is given the randomized puzzle now, he can decrypt the witness from the ciphertext using his private key and solve the randomized puzzle.

The complete protocol, excluding the registration phase described in Section 3.3.3, is illustrated in Figure 3.3.

Figure 3.3: A2L+ protocol.



Puzzle Promise. Bob initiates the protocol by requesting a pre-signature from the Hub. The Hub then chooses a uniformly sampled witness y (🔑) at random, generates a statement $Y := g^y$ from it, and computes the pSign algorithm with respect to some message m_{HB}, committing to the witness y. He transmits the pre-signature $\tilde{\sigma}_{HB}$, the encryption $c \leftarrow Enc(ek_H, y)$ of the witness y under his own encryption key and a non-interactive zero-knowledge proof (NIZK) π, that certifies that the ciphertext c encrypts the witness y, to Bob. Please note that in the figure, the lock icon 🔒 represents a tuple consisting of the statement and the ciphertext (Y, c).

Puzzle Transmission. Bob verifies that the pre-signature $\tilde{\sigma}_{HB}$ and the NIZK π are valid. If that is the case, he randomizes the statement Y and the ciphertext c to a fresh-looking version of the puzzle (🔒) and privately transmits them to Alice. Alice then randomizes the tuple again (🔒). Note that this second randomization would theoretically not be necessary. It is merely added to minimize trust assumptions between Alice and Bob.

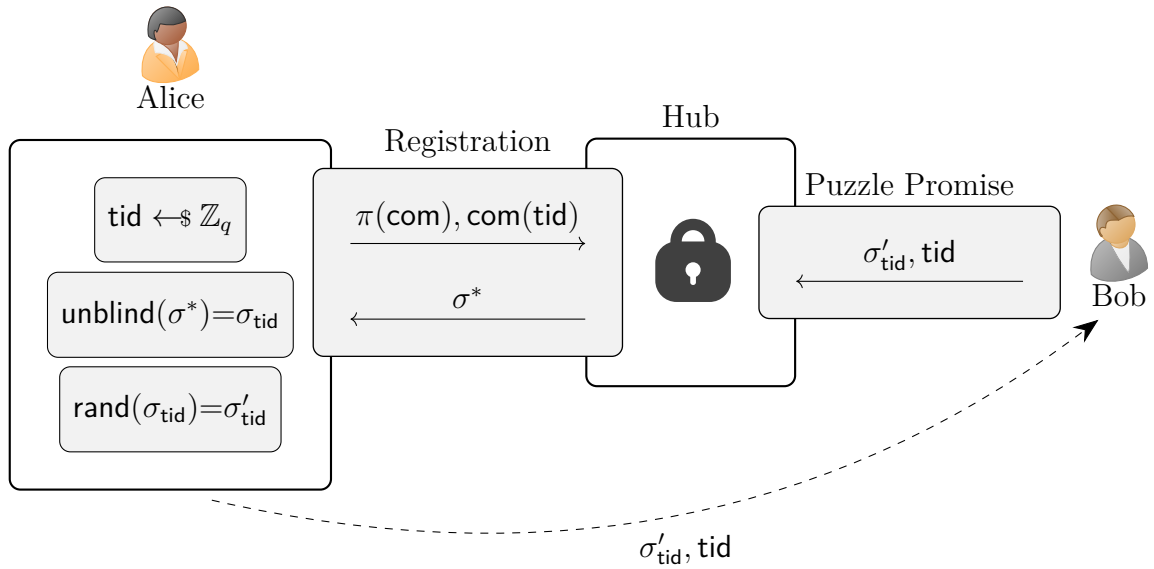
Puzzle Solver. Alice can now start the puzzle solver protocol. She creates a pre-signature $\tilde{\sigma}_{AH}$ with respect to some message m_{AH}, committing to the rerandomized puzzle (🔒) and thereby the rerandomized version of the original statement Y. She transmits the pre-signature $\tilde{\sigma}_{AH}$ alongside the rerandomized encryption of the witness to the Hub. The Hub can now use its decryption key to extract the witness from the ciphertext and solve the puzzle. He uses the witness to compute the Adapt algorithm and then publishes the full signature σ_{AH} to receive the transaction from Alice. Alice can now leverage the Extract algorithm to extract the witness 🔑 from the full signature σ_{AH} and the pre-signature $\tilde{\sigma}_{AH}$. Afterward, she derandomizes the witness 🔑 to the witness of the randomized puzzle she received from Bob and sends the solution 🔑 back to Bob. Bob finally derandomizes the witness again to receive the witness 🔑 to the original puzzle 🔒. With the witness 🔑 Bob can use the Adapt algorithm to create a full signature σ_{HB} on the message m_{HB} from the pre-signature $\tilde{\sigma}_{HB}$ and receive his coins.

3.3.3 Registration Protocol

The A2L+ protocol is initiated by the receiver Bob, requesting an initial commitment from the Hub in the form of a pre-signature. The Hub therefore needs to lock a certain amount of coins in each puzzle promise phase, without any assurance that the puzzle solver phase will be executed. This exposes the Hub to a griefing attack, where an adversarial receiver simply requests as many puzzle promises until all coins of the Hub are locked, effectively resulting in a denial of service attack.

To solve this, Tairi et al. [TMM21] proposed the registration protocol shown in Figure 3.4, which requires the sender to lock a certain amount of coins beforehand. The sender Alice first generates a random token identifier tid and a commitment com to it, using a Pedersen commitment, effectively locking the agreed amount of coins in

Figure 3.4: Registration protocol. Dashed arrows represent secure, private communication.



an escrow account. The commitment additionally ensures that she can recover the locked coins after a timeout period in case anything goes wrong. She then creates a NIZK proof π for the commitment and sends both the commitment com and the NIZK proof to the Hub. The Hub then validates the proof π, generates a blind signature σ^* on the token id tid, and sends it back to Alice. It is important that the tid is hidden inside the com and not visible to the Hub at this point. This is necessary since later the receiver Bob starts the puzzle promise protocol with the tid from Alice in the clear. If the tid is visible to the Hub here, linking Alice and Bob would be trivial for the Hub.

After Alice receives the blinded signature σ^*, she firstly unblinds it to receive the valid signature σ_{tid} and secondly randomizes this signature to a fresh-looking version σ'_{tid}. She then shares the valid signature σ'_{tid} and the tid privately with Bob. Bob uses $(\mathsf{tid}, \sigma'_{\mathsf{tid}})$ to prove to the Hub that some sender has locked the required coins to start the puzzle promise phase described in the previous Section 3.3.2.

As previously pointed out, the registration protocol is not needed in our blind conditional signature scheme alone. It should be emphasized however that it is indispensable for any coin-mixing protocol based on the presented concepts.

4 Definition and Current Security Notions of Blind Conditional Signatures

In this chapter, we will examine the current security notions of blind conditional signatures as defined by Glaeser et al. [GMM+22]. A2L+ is designed as a payment channel hub (PCH) based coin-mixing protocol. It is therefore assumed that a payment channel has already been established before the protocol starts. Blind conditional signatures were intended to serve as a new primitive to abstract coin-mixing and establish a new standard that can be proven secure under various settings. The following sections will introduce the game-based security notions by Glaeser et al. [GMM+22] and Tairi et al. [TMM21] in detail and pinpoint the important parts.

4.1 Definition

Blind conditional signatures (BCS) are executed among three parties referred to here as Alice, Bob, and the Hub. They are formally defined according to [GMM+22] as in Definition 19.

Definition 19 (Blind Conditional Signature [GMM+22])**.** *A blind conditional signature* $\Pi_{BCS} := (\mathit{Setup}, \mathit{PPromise}, \mathit{PSolver}, \mathit{Open})$ *is defined with respect to a signature scheme* $\Pi_{DS} := (\mathsf{KGen}, \mathsf{Sign}, \mathit{Vrf})$ *and consists of the following efficient algorithms:*

- $(\tilde{ek}, \tilde{dk}) \leftarrow \mathit{Setup}(1^\lambda)$*: The setup algorithm takes as input the security parameter* 1^λ *and outputs a key pair* $(\tilde{ek}, \tilde{dk})$*.*
- $(\perp, \{\tau, \perp\}) \leftarrow \mathit{PPromise}\left\langle \begin{matrix} H\left(\tilde{dk}, \mathsf{sk}^H, m_{HB}\right) \\ B\left(\tilde{ek}, \mathsf{vk}^H, m_{HB}\right) \end{matrix} \right\rangle$*: The puzzle promise algorithm is an interactive protocol between two users* H *(with inputs the decryption key* $\tilde{dk}$*, the signing key* sk^H*, and a message* m_{HB}*) and* B *(with inputs the encryption key* $\tilde{ek}$*, the verification key* vk^H*, and a message* m_{HB}*) and returns* $\perp$ *to* H *and either a puzzle* τ *or* $\perp$ *to* B*.*
- $(\{(\sigma^*, s), \perp\}, \{\sigma^*, \perp\}) \leftarrow \mathit{PSolver}\left\langle \begin{matrix} A\left(\mathsf{sk}^A, \tilde{ek}, m_{AH}, \tau\right) \\ H\left(\tilde{dk}, \mathsf{vk}^A, m_{AH}\right) \end{matrix} \right\rangle$*: The puzzle solving algorithm is an interactive protocol between two users* A *(with inputs the signing*

key sk^A*, the encryption key* $\tilde{\mathsf{ek}}$*, a message* m_{AH}*, and a puzzle* τ*) and* H *(with inputs the decryption key* $\tilde{\mathsf{dk}}$*, the verification key* vk^A*, and a message* m_{AH}*) and returns to both users either a signature* σ^* *(*A *additionally receives a secret* s*) or* $\perp$*.*

- $\{\sigma, \perp\} \leftarrow \mathit{Open}(\tau, s)$*: The open algorithm takes as input a puzzle* τ *and a secret* s *and returns a signature* σ *or* $\perp$*.*

Additionally, we define correctness as in Definition 20.

Definition 20 (Correctness [GMM+22])**.** *A blind conditional signature* Π_{BCS} *is correct if for all* $\lambda \in \mathbb{N}$*, all* $(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}})$ *in the support of* $\mathit{Setup}(1^\lambda)$*, all* $(\mathsf{vk}^H, \mathsf{sk}^H)$ *and* $(\mathsf{vk}^A, \mathsf{sk}^A)$ *in the support of* $\Pi_{\mathit{DS}}.\mathsf{KGen}(1^\lambda)$*, and all pairs of messages* (m_{HB}, m_{AH})*, it holds that*

$$\Pr\Big[\mathit{Vrf}(\mathsf{vk}^H, m_{HB}, \mathit{Open}(\tau, s)) = 1\Big] = 1$$

and

$$\Pr\Big[\mathit{Vrf}(\mathsf{vk}^A, m_{AH}, \sigma^*) = 1\Big] = 1$$

where

- $\tau \leftarrow \mathit{PPromise}\left\langle \begin{matrix} H\left(\tilde{\mathsf{dk}}, \mathsf{sk}^H, m_{HB}\right) \\ B\left(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB}\right) \end{matrix} \right\rangle$ *and*
- $((\sigma^*, s), \sigma^*) \leftarrow \mathit{PSolver}\left\langle \begin{matrix} A\left(\mathsf{sk}^A, \tilde{\mathsf{ek}}, m_{AH}, \tau\right) \\ H\left(\tilde{\mathsf{dk}}, \mathsf{vk}^A, m_{AH}\right) \end{matrix} \right\rangle$.

4.2 Blindness

Blindness informally states that the signer is not able to derive any information from the puzzle that can link the sender and the receiver. This means that there exists no PPT adversary that can link the two parties.

Definition 21 (Blindness [GMM+22])**.** *A blind conditional signature* Π_{BCS} *is blind if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for all* $\lambda \in \mathbb{N}$ *and all* PPT *adversaries* $\mathcal{A}$*, it holds that*

$$\Pr\Big[\mathit{ExpBlind}^{\mathcal{A}}_{\Pi_{\mathit{BCS}}}(\lambda) = 1\Big] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where $\mathit{ExpBlind}^{\mathcal{A}}_{\Pi_{\mathit{BCS}}}$ *is defined in Figure 4.1, and the probability is taken over the random choices of all probabilistic algorithms.*

Glaeser et al. [GMM+22] state that blindness is intuitively broken when the Hub and the receiver (Bob) collude. Therefore, they do not consider such a scenario and model only an adversarial Hub in the blindness experiment $\mathsf{ExpBlind}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ in Figure 4.1. We will later see the implications of this assumption on the blindness property of the protocol. For now, we only analyze the existing definition.

Figure 4.1: Blindness experiment.

$\mathsf{ExpBlind}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}(\lambda)$

1 : $(\tilde{\mathsf{ek}}, \mathsf{vk}_0^H, \mathsf{vk}_1^H, (m_{HB,0}, m_{AH,0}), (m_{HB,1}, m_{AH,1})) \leftarrow \mathcal{A}(1^\lambda)$
2 : $(\mathsf{vk}_0^A, \mathsf{sk}_0^A) \leftarrow \mathsf{KGen}(1^\lambda)$
3 : $(\mathsf{vk}_1^A, \mathsf{sk}_1^A) \leftarrow \mathsf{KGen}(1^\lambda)$
4 : $\tau_0 \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}_0^A, \mathsf{vk}_1^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}_0^H, m_{HB,0}) \Big\rangle$
5 : $\tau_1 \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}_0^A, \mathsf{vk}_1^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}_1^H, m_{HB,1}) \Big\rangle$
6 : $b \leftarrow \{0,1\}$
7 : $(\sigma_0^*, s_0) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}_0^A, \tilde{\mathsf{ek}}, m_{AH,0}, \tau_{0\oplus b}), \mathcal{A} \Big\rangle$
8 : $(\sigma_1^*, s_1) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}_1^A, \tilde{\mathsf{ek}}, m_{AH,1}, \tau_{1\oplus b}), \mathcal{A} \Big\rangle$
9 : **if** $(\sigma_0^* = \perp) \vee (\sigma_1^* = \perp) \vee (\tau_0 = \perp) \vee (\tau_1 = \perp)$
10 : $\quad \sigma_0 := \sigma_1 := \perp$
11 : **else**
12 : $\quad \sigma_{0\oplus b} \leftarrow \mathsf{Open}(\tau_{0\oplus b}, s_0)$
13 : $\quad \sigma_{1\oplus b} \leftarrow \mathsf{Open}(\tau_{1\oplus b}, s_1)$
14 : $b' \leftarrow \mathcal{A}(\sigma_0, \sigma_1)$
15 : **return** $(b = b')$

At the beginning of the given experiment the adversary on input the security parameter λ outputs an encryption key $\tilde{\mathsf{ek}}$ in support of an IND-CCA-secure homomorphic encryption scheme, two signing keys $\mathsf{vk}_0^H, \mathsf{vk}_1^H$, and two tuples consisting of two message pairs (m_{HB}, m_{AH}) for the puzzle promise and puzzle solver phases respectively. The adversarial Hub then performs the puzzle promise protocol with Bob and outputs two different puzzles τ_0 and τ_1. These two puzzles are then given to the adversary by Alice while performing the puzzle solver protocol, outputting two signature/secret tuples (σ^*, s). Finally, the Open algorithm is executed to output the signatures σ_0 and σ_1. The adversarial Hub wins the game if he can distinguish which signature belongs to which puzzle.

For blindness to hold, the puzzle can not leak any information that enables the adversary to link it to the underlying signature. Even if the Hub uses different signing keys for each puzzle promise and puzzle solver interaction, linking the two can not be possible. Intuitively, the A2L+ protocol achieves this by leveraging randomizable puzzles, namely by randomizing the embedded statements of the adaptor signatures.

4.3 Unlockability

The notion of unlockability ensures that it is hard for the Hub to generate a valid full signature during the puzzle solver protocol while not allowing Bob to create a valid full signature on the pre-signature of the corresponding puzzle promise. This evidently should prevent the Hub from stealing coins from Alice.

Definition 22 (Unlockability [GMM+22])**.** *A blind conditional signature* Π_{BCS} *is unlockable if there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for all* $\lambda \in \mathbb{N}$ *and all* PPT *adversaries* $\mathcal{A}$*, it holds that*

$$\Pr\left[\mathit{ExpUnlock}^{\mathcal{A}}_{\Pi_{BCS}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda),$$

where $\mathit{ExpUnlock}^{\mathcal{A}}_{\Pi_{BCS}}$ *is defined in Figure 4.2, and the probability is taken over the random choices of all probabilistic algorithms.*

Figure 4.2: Unlockability experiment.

$\mathsf{ExpUnlock}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}(\lambda)$

1 : $(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB}, m_{AH}) \leftarrow \mathcal{A}(1^\lambda)$
2 : $(\mathsf{vk}^A, \mathsf{sk}^A) \leftarrow \mathsf{KGen}(1^\lambda)$
3 : $\tau \leftarrow \mathsf{PPromise}\langle \mathcal{A}(\mathsf{vk}^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB})\rangle$
4 : **if** $\tau = \perp$
5 : $\quad(\hat{\sigma}, \hat{m}) \leftarrow \mathcal{A}$
6 : $\quad b_0 := (\mathsf{Vf}(\mathsf{vk}^A, \hat{\sigma}, \hat{m}) = 1)$
7 : **if** $\tau \neq \perp$
8 : $\quad(\sigma^*, s) \leftarrow \mathsf{PSolver}\langle A(\mathsf{sk}^A, \tilde{\mathsf{ek}}, m_{AH}, \tau), \mathcal{A}\rangle$
9 : $\quad(\hat{\sigma}, \hat{m}) \leftarrow \mathcal{A}$
10 : $\quad b_1 := (\mathsf{Vf}(\mathsf{vk}^A, \hat{\sigma}, \hat{m}) = 1) \wedge (\hat{m} \neq m_{AH}))$
11 : $\quad b_2 := (\mathsf{Vf}(\mathsf{vk}^A, \sigma^*, m_{AH}) = 1)$
12 : $\quad b_3 := (\mathsf{Vf}(\mathsf{vk}^H, m_{HB}, \mathsf{Open}(\tau, s)) \neq 1)$
13 : **return** $b_0 \vee b_1 \vee (b_2 \wedge b_3)$

In the unlockability experiment $\mathsf{ExpUnlock}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ in Figure 4.2 again, an adversarial Hub is modeled. On input the security parameter λ, the adversary outputs the encryption key $\tilde{\mathsf{ek}}$, a signing key vk^H, and two messages m_{HB} and m_{AH} for the puzzle promise and puzzle solver phases respectively. Additionally, a signing/verification key pair is generated by the sender Alice $(\mathsf{vk}^A, \mathsf{sk}^A)$. The Hub and Bob then engage in the puzzle promise protocol and output a puzzle τ. If τ is valid, the Hub engages in the puzzle solver protocol with Alice, outputting a signature σ^* and a secret s.

The adversary wins the game if it can fulfill one of the following three conditions:

- b_0: Output a valid message signature pair under Alice's verification key vk^A, while τ is invalid and therefore does not allow extracting a valid signature on the message m_{HB}.
- b_1: Output a valid message signature pair signed under Alice's verification key vk^A on a message other than m_{AH}, effectively creating a valid forgery and stealing coins from Alice.
- $b_2 \wedge b_3$: Output a valid signature for the message m_{AH} signed under Alice's verification key vk^A, while the secret s from the puzzle solver protocol does not allow Bob to output a valid signature on the message m_{HB} signed by the Hub.

In the unlockability experiment $\mathsf{ExpBlnd}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ in Figure 4.2, the focus is on enabling Bob to generate a valid full signature for the previously agreed upon message m_{HB}. In a blockchain-based setting, such a message might correspond to a transaction. Apart from generating a valid signature for the given transaction, a few other conditions need to hold for the transaction to be accepted e.g., the transaction inputs need to be unspent. The A2L+ protocol solves this by requiring a payment channel setup before the protocol execution is initiated, effectively rendering such a scenario impossible.

4.4 Unforgeability

Generally speaking, the goal of the notion of unforgeability is to prevent the adversary from forging a signature on any message with a signing key other than its own. Unforgeability therefore protects the signer and guarantees that any valid signature has been signed by no other user than the one holding the signing key. In any blockchain based setting this ensures the security of the coins, since transaction messages can only be signed by the owner of the secret signing key. When using blind conditional signatures as a coin-mixing protocol, the notion of unforgeability more precisely prevents anyone but the Hub from signing transaction messages under the Hub's verification key vk^H and therefore protects the Hub from getting coins stolen. In contrast to the preceding security games, Glaeser et al. [GMM+22] give the adversary access to a puzzle promise and a puzzle solver oracle in Definition 23.

Definition 23 (Unforgeability [GMM+22])**.** *A blind conditional signature Π_{BCS} is unforgeable if there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$ and all PPT adversaries $\mathcal{A}$, it holds that*

$$\Pr\left[\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda),$$

where $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ is defined in Figure 4.3, and the probability is taken over the random choices of all probabilistic algorithms.

In the unforgeability experiment $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ in Figure 4.3 on input the security parameter λ a key pair $(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}})$ in support of the IND-CCA-secure homomorphic

Figure 4.3: Unforgeability experiment.

$\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}(\lambda)$

1 : $\mathcal{L} := \emptyset, \mathcal{Q} := 0$
2 : $(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}}) \leftarrow \mathsf{Setup}(1^\lambda)$
3 : $(\mathsf{vk}_1^H, m_1, \sigma_1), ..., (\mathsf{vk}_q^H, m_q, \sigma_q) \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{PP}(\cdot),\mathcal{O}\mathsf{PS}(\cdot)}(\tilde{\mathsf{ek}})$
4 : $b_0 := \exists i \in [q]\ s.t.\ (\mathsf{vk}_i^H, \cdot) \in \mathcal{L} \wedge (\mathsf{vk}_i^H, m_i) \notin \mathcal{L} \wedge \mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1$
5 : $b_1 := \forall i \in [q],\ (\mathsf{vk}_i^H, m_i) \in \mathcal{L} \wedge \mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1$
6 : $b_2 := \bigwedge_{i,j \in [q], i \neq j} (\mathsf{vk}_i^H, m_i, \sigma_i) \neq (\mathsf{vk}_j^H, m_j, \sigma_j)$
7 : $b_3 := (\mathcal{Q} \leq q - 1)$
8 : **return** $b_0 \vee (b_1 \wedge b_2 \wedge b_3)$

$\mathcal{O}\mathsf{PP}(m)$

1 : $(\mathsf{vk}^H, \mathsf{sk}^H) \leftarrow \Pi_{\mathsf{AS}}.\mathsf{KGen}(1^\lambda)$
2 : $\mathcal{L} := \mathcal{L} \cup \{(\mathsf{vk}^H, m)\}$
3 : $\perp \leftarrow \mathsf{PPromise}\Big\langle H(\tilde{\mathsf{dk}}, \mathsf{sk}^H, m), \mathcal{A}(\mathsf{vk}^H)\Big\rangle$

$\mathcal{O}\mathsf{PS}(\mathsf{vk}^A, m')$

1 : $(\sigma^*, s) \leftarrow \mathsf{PSolver}\Big\langle \mathcal{A}, H(\tilde{\mathsf{dk}}, \mathsf{vk}^A, m')\Big\rangle$
2 : **if** $\sigma^* \neq \perp$ **then** $\mathcal{Q} := \mathcal{Q} + 1$

encryption scheme Π_{E} according to Section 2.3 is sampled. The adversary $\mathcal{A}$ then generates q triples $(\mathsf{vk}^H, m, \sigma)$ by querying the puzzle promise and puzzle solver oracles $\mathcal{O}\mathsf{PP}$ and $\mathcal{O}\mathsf{PS}$. The adversary wins the game by either outputting a valid forgery, meaning a signature on a previously not queried message (condition b_0), or if all the below conditions hold:

- b_1: All q triples $(\mathsf{vk}^H, m, \sigma)$ are valid and the puzzle promise oracle $\mathcal{O}\mathsf{PP}$ has been queried for all q corresponding verification key, message pairs (vk, m). This ensures that all signatures are valid and signed under a verification key vk_i^H owned by the Hub.
- b_2: All q triples $(\mathsf{vk}^H, m, \sigma)$ are different, meaning the adversary was able to output q distinct signatures.
- b_3: The puzzle solver oracle $\mathcal{O}\mathsf{PS}$ has been queried at most $q-1$ times. Therefore, at least one signature was not honestly signed by the Hub and needs to be a forgery created by the adversary.

The puzzle promise and puzzle solver oracles $\mathcal{O}\mathsf{PP}$ and $\mathcal{O}\mathsf{PS}$ are modeled in Figure 4.3 in an idealized way. On each query to $\mathcal{O}\mathsf{PP}(m)$, a new key pair $(\mathsf{vk}^H, \mathsf{sk}^H)$ in support of the key generation algorithm KGen of the adaptor signature scheme Π_{AS} is sampled and the verification key/message tuple (vk^H, m) is added to the language $\mathcal{L}$. The puzzle solver oracle $\mathcal{O}\mathsf{PS}(\mathsf{vk}^A, m')$ on input Alice's verification key vk^A and a message m' runs the $\mathsf{PSolver}$ protocol. If the $\mathsf{PSolver}$ execution was successful, $\mathcal{O}\mathsf{PS}$ returns the resulting signature σ^* alongside the secret s. Additionally, it increments the query counter $\mathcal{Q}$ by one.

Important to note here is that the secret s output by the $\mathcal{O}\mathsf{PS}$ oracle is not necessarily bound to a single message signature tuple. Even though condition b_2 states that all tuples need to be different, this does not imply that all embedded secrets must also be different. This distinction will become crucial in the following chapters.

4.5 One-More CCA-A2L Security

Glaser et al. [GMM+22] introduced the security notion of *one-more CCA-A2L security (OM-CCA-A2L)* as a helper for their security proof of unforgeability. Let Π_{E} be a IND-CCA-secure homomorphic public key encryption scheme according to Section 2.3. In the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}$ experiment first the query counter $\mathcal{Q}$ is set to 0 and on input the security parameter λ a key pair $(\mathsf{ek}, \mathsf{dk})$ is generated using the KGen algorithm of the encryption scheme Π_{E}. Then again, on input of the security parameter λ, several $q+1$ random values $r_1, \ldots, r_{q+1}$ are generated and encrypted using the Enc algorithm. The adversary is given the encryption key ek, the encrypted values c_i, and the statements of the original random values $h_i = g^{r_i}$. Additionally, an oracle $\mathcal{O}$ is provided that takes as inputs a verification key vk, message m, statement h, encrypted value c, and pre-signature $\tilde{\sigma}$. $\mathcal{O}$ returns an adapted full signature on the message m if the pre-verification algorithm of the adaptor signature scheme Π_{AS} verifies the correctness of the pre-signature $\tilde{\sigma}$ and the value encrypted in the ciphertext c is a valid witness to the statement h, otherwise it returns $\perp$. The adversary wins the game if it can output $q+1$ values r'_i, such that for all $i \in \{1, \ldots, q+1\}$ it holds that $r_i = r'_i$ and the oracle $\mathcal{O}$ has been queried at most q times.

Definition 24 (OM-CCA-A2L [GMM+22])**.** *An encryption scheme Π_{E} is one-more CCA-A2L-secure (OM-CCA-A2L) if there exists a negligible function $\mathsf{negl}(\lambda)$ such that for all $\lambda \in \mathbb{N}$, all polynomials $q = q(\lambda)$, and all PPT adversaries $\mathcal{A}$, it holds that*

$$\Pr\left[\mathit{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda),$$

where $\mathit{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}$ is defined in Figure 4.4, and the probability is taken over the random choices of all probabilistic algorithms.

The $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}$ security game serves as an adaptation of the one-more discrete logarithm assumption. The adversary aims to find the witnesses r_i to the statements $h_i = g^{r_i}$. Given that the encryption scheme is IND-CCA-secure and the adversary

Figure 4.4: OM-CCA experiment.

$\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_\mathsf{E},q}(\lambda)$

1 : $\mathcal{Q} := 0$
2 : $(\mathsf{ek}, \mathsf{dk}) \leftarrow \Pi_\mathsf{E}.\mathsf{KGen}(1^\lambda)$
3 : $r_1, \ldots, r_{q+1} \leftarrow\!\!\$\ \{0,1\}^\lambda$
4 : $c_i \leftarrow \Pi_\mathsf{E}.\mathsf{Enc}(\mathsf{ek}, r_i)$
5 : $(r'_1, \ldots, r'_{q+1}) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{A^2L}}_{\mathsf{dk},\Pi_\mathsf{E},\Pi_\mathsf{AS}}}(\mathsf{ek}, (c_1, g^{r_1}), \ldots, (c_{q+1}, g^{r_{q+1}}))$
6 : **if** $r'_i = r_i\ \forall i \in 1, \ldots, q+1 \wedge \mathcal{Q} \leq q$ **then return** 1
7 : **else return** 0

$\mathcal{O}^{\mathsf{A^2L}}_{\mathsf{dk},\Pi_\mathsf{E},\Pi_\mathsf{AS}}(\mathsf{vk}, m, h, c, \widetilde{\sigma})$

1 : **check if** $\mathsf{vk} \in \mathsf{Supp}(\Pi_\mathsf{AS}.\mathsf{KGen}(1^\lambda))$
2 : $\widetilde{x} \leftarrow \Pi_\mathsf{E}.\mathsf{Dec}(\mathsf{dk}, c)$
3 : **if** $\Pi_\mathsf{AS}.\mathsf{pVrf}(\mathsf{vk}, m, h, \widetilde{\sigma}) = 1$ **and** $g^{\tilde{x}} = h$
4 : $\quad \mathcal{Q} := \mathcal{Q} + 1$
5 : $\quad$ **return** $\sigma' \leftarrow \Pi_\mathsf{AS}.\mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}, \widetilde{x})$
6 : **else return** $\perp$

$\mathcal{A}$ does not know the decryption key dk, the probability of outputting the decrypted witnesses r_i given the ciphertexts c_i is negligible without access to an oracle. Therefore, outputting the witness r_i for any statement h_i is indistinguishable from solving the discrete logarithm. It holds that winning the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_\mathsf{E},q}$ game is computationally indistinguishable from solving the one-more discrete logarithm assumption, which is believed to be hard in polynomial time. The formal proof can be found in the work by Glaeser et al. [GMM+22].

In contrast to the unforgeability experiment $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_\mathsf{BCS}}$ in Figure 4.3, the adversary in the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_\mathsf{E},q}$ experiment needs to find the witnesses to the encrypted statements and not the signatures of messages under a verification key vk. More specifically, in the $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_\mathsf{BCS}}$ game, the goal of $\mathcal{A}$ is to output q valid signatures while querying the $\mathcal{O}\mathsf{PS}$ oracle at most $q-1$ times. This does not inherently imply that all secrets s output by the $\mathsf{PSolver}$ algorithm during the $\mathcal{O}\mathsf{PS}$ queries are different. In contrast to the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_\mathsf{E},q}$ game, winning the $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_\mathsf{BCS}}$ game therefore does not imply that q unique witnesses to the embedded statements have been output. Evidently, the hardness of the underlying mathematical problems differs. This divergence highlights the need to exercise caution when attempting to reduce the unforgeability experiment $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_\mathsf{BCS}}$ to the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_\mathsf{E},q}$ experiment.

5 Problems with Current Security Notions

In this chapter, we identify shortcomings of the current security definitions of blind conditional signatures and issues with the A2L+ protocol construction by Glaeser et al. [GMM+22]. Some of which stem from the security properties of adaptor signatures by Aumayr et al. [AEE+21], which have limitations outside a payment channel setting, as has been pointed out by [GSST24, DOY22]. These limited security properties introduced a flaw in the security proof of unforgeability, as we will see in Section 5.2. Other issues arise from system assumptions that do not always hold and render the scheme insecure under various conditions. Finally, we examine the limitations associated with the payment channel setup of the A2L+ protocol.

5.1 Problems with Current Blindness Definition

In the A2L+ protocol, the authors assume that the Hub does not collude with either the sender (Alice) or the receiver (Bob). It is furthermore stated that in case of a collusion, linking two transactions becomes trivial since the other party can simply reveal the identity to the Hub. We boldly state that this assumption is far from a real-world scenario. Say Alice communicates with Bob over a secure private channel. The only information Alice or Bob might have about each other is the IP address of the network from which they are accessing the internet. Both of them however may still want to hide their public blockchain addresses from each other and the Hub. Effectively, even when the Hub colludes with either one of the two, the only information that must be leaked are the IP addresses and any auxiliary information that the parties willingly share. Our protocol therefore needs to protect the sender and the receiver from getting deanonymized in case of collusion.

Additionally, Tairi et al. [TMM21] acknowledge that two transactions can be easily linked when the Hub reduces the anonymity set. This weakness is unfortunately present in all coin-mixing applications. To circumvent that, the parties need to check the anonymity set themselves after the protocol execution finishes. To improve anonymity, they can further engage in more rounds of the protocol.

First, we will analyze the impact of aborts on the blindness property. Following this, we will demonstrate how the blindness of the A2L+ protocol can be compromised through collusion between the Hub and one of the other parties.

5.1.1 Blindness under Aborts

First, we investigate blindness under aborts in which the Hub acts as the malicious party. As stated by Tairi et al. [TMM21], the Hub alone can reduce the anonymity set and effectively break blindness by letting all but one transaction fail in a single epoch. While the size of the anonymity set can be checked at the end of each epoch, we still need to ensure that the adversary can not deduce any information from letting an execution fail.

In the blindness experiment $\mathsf{ExpBlnd}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ in Figure 4.1 in Lines 9 and 10, the adversary receives $\perp$ for both signatures when one of the puzzles or signatures is invalid ($\perp$):

$$\begin{aligned}&\textbf{if } ((\sigma_0^* = \perp) \vee (\sigma_1^* = \perp) \vee (\tau_0 = \perp) \vee (\tau_1 = \perp))\\ &\quad \sigma_0 := \sigma_1 := \perp\end{aligned}$$

In the A2L+ protocol, this is however not exactly the case. When receiving $\perp$ on one of the puzzles/signatures, we do not get $\perp$ for all the other executions as well. By simply letting one of the puzzle promise $\mathsf{PPromise}$ or puzzle solver $\mathsf{PSolver}$ protocols return $\perp$, the adversary additionally learns which execution failed and which succeeded. To model this more precisely, we introduce the notion of selective failure blindness in Section 6.1.

5.1.2 Blindness Between Sender and Receiving Parties

In this section, we explore a weakness of the A2L+ protocol, which can only be exploited when the Hub and Bob collude. We argue that Alice at least needs to be able to detect that blindness is broken in case there is a collusion or the $\mathsf{PSolver}$ execution needs to fail. The origin of this weakness lies in the fact that Alice fully trusts Bob regarding the validity of the puzzle. Since Alice also locks her coins in the pre-signature $\tilde{\sigma}_{AH}$ based on the statement Y'' contained in the puzzle, we further argue that she needs to be able to prove in some way that the puzzle she receives from Bob is valid.

Let the Hub sample a set of key pairs, each consisting of an encryption key and a decryption key $(\mathsf{ek}, \mathsf{dk})$. For each puzzle promise protocol, the Hub now chooses a different encryption key ek from the set to generate the ciphertext of the witness for the associated challenge alongside the NIZK π. When checking the validity of the NIZK π, an honest Bob would detect that the used encryption key ek is not the same as the advertised public key of the Hub. Since Bob is acting maliciously, he simply sends the ciphertext and the challenge along to Alice. Alice now randomizes the ciphertext and initiates the puzzle solver protocol. The only thing left to do for the Hub is to try decrypting the ciphertext with every decryption key previously used. As soon as he finds a matching key, he can link Alice's public address to Bob.

Note that in our Construction 1 of a randomizable puzzle scheme, in the PRand algorithm, the public key is required to generate a randomized puzzle that can be solved using the public key of the Hub as a trapdoor. Hence, if the public key used to generate the original puzzle is different from the advertised public key of the Hub, the $\mathsf{PSolver}$

protocol will fail. Unfortunately, not all randomizable puzzle constructions have this inherent property and the issue that Alice's coins are locked for the duration of the protocol execution still persists. To combat this, we implicitly include a randomized NIZK into the output of the PPromise protocol, which Alice needs to verify during the PSolver execution in our revised Construction 3 of the A2L+ protocol.

According to the system assumptions, the $\mathsf{ExpBlnd}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ assumes that there is no collusion and hence the game is correctly modeled in that regard. This however highlights the complexity of proofing the security of a coin-mixing protocol under various circumstances.

5.1.3 Blindness Between Sending Parties and Receiver

Similar to the above, we now examine the case of collusion between the sender (Alice) and the Hub. The registration protocol introduces a weakness in the protocol, which can be used to break blindness. As previously mentioned, the registration protocol is an extension to blind conditional signatures and is needed to protect against griefing attacks. This type of attack is mostly beneficial when blind conditional signatures are used as a coin-mixing protocol, where monetary assets are being transferred. When looking at the blindness experiment, the registration protocol can be seen as part of the puzzle promise phase.

During the registration protocol described in Section 3.3.3, Alice first chooses a random transaction tid and creates a commitment com to it, which gets blindly signed by the Hub. This signed commitment com is later used to initiate the puzzle promise protocol and serves as proof of collateral. To prove to the Hub that the collateral has been locked, Bob needs to send the tid in the clear at the beginning of the puzzle promise protocol alongside the unblinded and randomized signature on the tid. Since the Hub and Alice collude, they can easily link the two transactions by checking the used tid. In a scenario where a collusion between Alice and the Hub must be prevented, the receiver Bob therefore must execute the registration protocol himself. Since this is not an issue with blind conditional signatures, we will not further investigate it in this work.

5.2 Problems with Current Unforgeability Definition

As has been previously mentioned, the security definitions of adaptor signatures by Aumayr et al. [AEE+21] have some limitations, which have first been discovered by Dai et al. [DOY22]. These limitations mostly stem from system assumptions that hold in the context of the original definitions and their purpose. For our use case, these assumptions do however not always hold and therefore lead to an unsecure protocol. Unfortunately, Tairi et al. [TMM21] and Glaeser et al. [GMM+22] still rely on the definitions by Aumayr et al. [AEE+21]. This introduces a weakness into BCS and allows breaking unforgeability.

The following attack has been introduced by Gerhart et al. [GSST24]. We first recall

their construction of a secure adaptor signature scheme according to the definitions by Aumayr et al. [AEE+21], which has malleable pre-signatures in figure Figure 5.1.

Figure 5.1: Adaptor signature scheme Π'_{AS} with malleable pre-signatures.

$\mathsf{pSign}'(\mathsf{sk}, m, Y)$

1 : $(r_1, r_2) \leftarrow\!\$\ \mathbb{Z}_p^2$
2 : $\widetilde{\sigma}_1 \leftarrow \mathsf{pSign}(\mathsf{sk}, m, Y; r_1)$
3 : $\widetilde{\sigma}_2 \leftarrow \mathsf{pSign}(\mathsf{sk}, m, Y; r_2)$
4 : **return** $(\widetilde{\sigma}_1, \widetilde{\sigma}_2)$

$\mathsf{pVrf}'(\mathsf{vk}, m, Y, \widetilde{\sigma})$

1 : $(\widetilde{\sigma}_1, \widetilde{\sigma}_2) =: \widetilde{\sigma}$
2 : $b_1 := \mathsf{pVrf}(\mathsf{vk}, m, Y, \widetilde{\sigma}_1)$
3 : $b_2 := \mathsf{pVrf}(\mathsf{vk}, m, Y, \widetilde{\sigma}_2)$
4 : **return** $b_1 \wedge b_2$

$\mathsf{Adapt}'(\mathsf{vk}, \widetilde{\sigma}, y)$

1 : $(\widetilde{\sigma}_1, \widetilde{\sigma}_2) =: \widetilde{\sigma}$
2 : $\sigma := \mathsf{Adapt}(\mathsf{vk}, \widetilde{\sigma}_1, y)$
3 : **return** σ

$\mathsf{Extract}'(\mathsf{vk}, \widetilde{\sigma}, \sigma, Y)$

1 : $(\widetilde{\sigma}_1, \widetilde{\sigma}_2) =: \widetilde{\sigma}$
2 : $y =: \mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}_1, \sigma, Y)$
3 : **if** $(Y, y) \in \mathsf{Rel}$ **return** y
4 : **return** $\mathsf{Extract}(\mathsf{vk}, \widetilde{\sigma}_2, \sigma, Y)$

The adaptor signature scheme Π'_{AS} is based on an unconnected adaptor signature scheme Π_{AS}. Each pre-signature consists of two distinct pre-signatures on the same message and statement. This allows us to adapt each pre-signature to two distinct full signature on the same message.

We now additionally recall the adversary $\mathcal{A}$ according to Gerhart et al. [GSST24] that breaks unforgeability as defined in Figure 4.3, when instantiated with the adaptor signature scheme Π'_{AS}. Let the adversary $\mathcal{A}$ query the puzzle promise and puzzle solver oracles $(\mathcal{O}\mathsf{PP}, \mathcal{O}\mathsf{PS})$ at least one time each on a random message m to learn a pre-signature $\widetilde{\sigma}$ and a full signature σ on the message m and the random statement Y. The adversary now queries the $\mathsf{Extract}$ algorithm using $\widetilde{\sigma}$ and σ to learn the witness y to the statement Y. With the pre-signature $\widetilde{\sigma}$ and the corresponding witness y, $\mathcal{A}$ parses $\widetilde{\sigma} := (\widetilde{\sigma}_1, \widetilde{\sigma}_2)$ and computes $\widetilde{\sigma}' := (\widetilde{\sigma}_2, \widetilde{\sigma}_1)$, by swapping the order of the pair. Finally, the adversary queries the Adapt' algorithm on input $(\widetilde{\sigma}')$ and learns a second full signature σ' on the message m. Since Π_{AS} is an unconnected adaptor signature scheme, σ' is distinct from σ with overwhelming probability.

With this attack, the adversary renders the conditions b_1, b_2 and b_3 in Lines 5 to 7 of $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ in Figure 4.3 to true and therefore breaks unforgeability by winning the game. More specifically, each verification key, message pair (vk, m) is in the language $\mathcal{L}$, and each σ is a valid signature on m (b_1). All triples of (vk, m, σ) are unique (b_2) since all the signatures σ are distinct and the puzzle solver oracle $\mathcal{O}\mathsf{PS}$ has been queried at least one time less than valid signatures output (b_3).

To prevent this attack, we require that the adaptor signature scheme Π_{AS} in our revised version of the A2L+ protocol in Construction 3 satisfies the improved security definitions according to Gerhart et al. [GSST24] as stated in Section 2.8.2.

5.3 Limitations of Payment Channel Setup

One of the goals of the A2L+ protocol was to design a highly performant and widely compatible coin-mixing protocol. To solve that, Glaeser et al. [GMM+22] introduced the cryptographic concept of blind conditional signatures. This new concept only requires adaptor signatures and a NIZK proof system as an underlying cryptographic primitive. When constructing a coin-mixing protocol and instead of signing arbitrary messages we sign blockchain transactions, the situation however changes. In the puzzle promise phase, the Hub now creates a pre-signature on a transaction. This pre-signature $\tilde{\sigma}$ and the encapsulated statement Y essentially certify that given a witness y to the statement Y, the receiver can complete the pre-signature $\tilde{\sigma}$ to a full signature on the transaction. This allows to retrieve the previously agreed-upon coins. Since the puzzle solver protocol gets executed at a later point in time and the receiver can only retrieve their coins from the Hub, once they receive the witness y to the statement Y, it needs to be ensured that the Hub does not spend the coins in the meantime. Given the nature of blockchains, we therefore need a method to lock the transaction inputs of the Hub and give the receiver enough time to publish the transaction on-chain and retrieve the coins. In A2L+, this is done implicitly by requiring a payment channel setup before the protocol starts. This setup inherently does not allow one party to act maliciously without getting punished. Even though this does solve our problem at hand, a payment channel setup is quite a complex system and might not always be realizable. As mentioned at the beginning of this section, we are aiming at a widely compatible coin-mixing protocol that may even support cross-chain transactions. Having the payment channel setup as a prerequisite therefore contradicts one of our main goals. We therefore leave it open to choose the preferred method of implementation and simply emphasize here that the coins need to be locked for the duration of the protocol execution.

6 Enhanced Security Definitions

In the previous chapter, we elaborated on the shortcomings of the current security notions. We will now introduce the notion of selective failure blindness, a strictly stronger security property for blindness. Selective failure blindness ensures blindness even in the case of adversarial aborts.

6.1 Selective-Failure Blindness

Camenisch et al. [CNs07] first introduced the notion of selective-failure blindness, which states that blindness also needs to hold in case the signer learns that some executions failed. While investigating the security of blind signatures under aborts, Fischlin et al. [FS09] augmented the work of Camenisch et al. [CNs07] and showed how every blind signature scheme can be turned into one that is selective-failure blind.

In this section, we will first restate the general notion of selective failure-blindness in Definition 25 and then present an adapted definition for a selective-failure blind conditional signature scheme in Definition 26.

Definition 25 (Selective-Failure Blindness [CNs07])**.** *A blind signature scheme* $\Pi_{BS} = (\mathsf{KGen}_{BS}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathit{Vrf}_{BS})$ *is selective-failure blind if it is strongly unforgeable according to Definition 10 and for every* PPT *adversary* $\mathcal{A}$ *(working in modes find, issue and guess) there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for every* $\lambda \in \mathbb{N}$

$$\Pr\left[\mathit{ExpSFBlnd}^{\mathcal{A}}_{\Pi_{BS}}(\lambda) = 1\right] \leq 1/2 + \mathsf{negl}(\lambda),$$

where experiment $\mathit{ExpSFBlnd}^{\mathcal{A}}_{\Pi_{BS}}$ *is described in Figure 6.1, and the probability is taken over the random choices of all probabilistic algorithms.*

Camenisch et al. [CNs07] constructed the adversary $\mathcal{A}$ in Definition 25 to work in three modes: find, issue, and guess. The find mode allows $\mathcal{A}$ to identify and set up the necessary inputs for the attack, such as generating the key pair and messages. In the issue mode, $\mathcal{A}$ interacts with the honest user algorithm $\mathcal{U}$ to gather further information for outputting the signatures σ. Finally, the guess mode involves the adversary making its final guess about the value of bit b. Our revised version of selective-failure blindness for blind conditional signatures in Figure 6.2 does not use these modes but follows a different interaction structure.

In the $\mathsf{ExpSFBlnd}^{\mathcal{A}}_{\Pi_{\mathsf{BS}}}(\lambda)$ experiment in Figure 6.1 instead of simply returning $\perp$ in case a signature execution fails, the adversarial signer is additionally given the information which instance was aborted. A scheme is selective-failure blind if an

Figure 6.1: Selective failure blindness experiment.

$\mathsf{ExpSFBlnd}^{\mathcal{A}}_{\Pi_{\mathsf{BS}}}(\lambda)$

1 : $(\mathsf{pk}_{BS}, m_0, m_1, \beta_{find}) \leftarrow \mathcal{A}(find, 1^\lambda)$
2 : $b \leftarrow \{0,1\}$
3 : $\beta_{issue} \leftarrow \mathcal{A}^{\langle\cdot,\mathcal{U}(\mathsf{pk}_{BS},m_b)\rangle^1,\langle\cdot,\mathcal{U}(\mathsf{pk}_{BS},m_{1-b})\rangle^1}(issue, \beta_{find})$
4 : and let σ_b, σ_{1-b} denote the (possibly undefined) local outputs
5 : of $\mathcal{U}(\mathsf{pk}_{BS}, m_b)$ resp. $\mathcal{U}(\mathsf{pk}_{BS}, m_{1-b})$.
6 : define answer as:
7 : left if only the first execution has failed,
8 : right if only the second execution has failed,
9 : both if both executions have failed,
10 : and (σ_b, σ_{1-b}) otherwise.
11 : $b^* \leftarrow \mathcal{A}(guess, answer, \beta_{issue})$
12 : **return** 1 iff $b = b^*$

adversarial signer cannot determine with a probability significantly greater than 1/2 which signature belongs to which message.

Definition 26 (Selective-Failure Blindness)**.** *A blind conditional signature scheme* Π_{BCS} *is selective-failure blind if it is strongly unforgeable according to Definition 10 and for every* PPT *adversary* $\mathcal{A}$ *there exists a negligible function* $\mathsf{negl}(\lambda)$ *such that for every* $\lambda \in \mathbb{N}$

$$\Pr\left[\mathit{ExpSFBlnd}^{\mathcal{A}}_{\Pi_{BCS}}(\lambda) = 1\right] \leq 1/2 + \mathsf{negl}(\lambda),$$

where experiment $\mathit{ExpSFBlnd}^{\mathcal{A}}_{\Pi_{BCS}}$ *is described in Figure 6.2, and the probability is taken over the random choices of all probabilistic algorithms.*

Building on this concept, we present an adapted definition for a selective-failure blind conditional signature scheme in Definition 26. With this improved definition, the adversary cannot distinguish between two signatures, even if it knows which of the two puzzle solver executions failed. More precisely, rather than simply returning $\perp$ when any of the puzzle solver PSolver executions fail, the experiment identifies and returns the specific executions that failed. Furthermore, when one of the puzzle promise PPromise executions fails, the protocol aborts immediately. This is necessary since otherwise, it becomes trivial for any adversary to link the PPromise and PSolver executions and win the game. We want to stress here that in a real-world scenario, an adversarial Hub would learn the input to the PSolver protocol (vk^A, m_{AH}) only after a successful execution of the corresponding PPromise protocol. Therefore, to break selective-failure blindness, the adversary would need to reduce the anonymity set to one and abort all other PPromise executions in an epoch. Since the information from the input of the PPromise protocol alone does not allow the Hub to make an

Figure 6.2: Selective failure blindness for blind conditional signatures experiment.

$\mathsf{ExpSFBlind}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}(\lambda)$

1 : $(\tilde{\mathsf{ek}}, \mathsf{vk}_0^H, \mathsf{vk}_1^H, (m_{HB,0}, m_{AH,0}), (m_{HB,1}, m_{AH,1})) \leftarrow \mathcal{A}(1^\lambda)$
2 : $(\mathsf{vk}_0^A, \mathsf{sk}_0^A) \leftarrow \mathsf{KGen}(1^\lambda)$
3 : $(\mathsf{vk}_1^A, \mathsf{sk}_1^A) \leftarrow \mathsf{KGen}(1^\lambda)$
4 : $\tau_0 \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}_0^A, \mathsf{vk}_1^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}_0^H, m_{HB,0}) \Big\rangle$
5 : $\tau_1 \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}_0^A, \mathsf{vk}_1^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}_1^H, m_{HB,1}) \Big\rangle$
6 : **if** $(\tau_0 = \perp) \vee (\tau_1 = \perp)$ **then return** 0
7 : $b \leftarrow \{0,1\}$
8 : $(\sigma_0^*, s_0) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}_0^A, \tilde{\mathsf{ek}}, m_{AH,0}, \tau_{0\oplus b}), \mathcal{A} \Big\rangle$
9 : $(\sigma_1^*, s_1) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}_1^A, \tilde{\mathsf{ek}}, m_{AH,1}, \tau_{1\oplus b}), \mathcal{A} \Big\rangle$
10 : **if** $(\sigma_0^* = \perp) \vee (\sigma_1^* = \perp)$ define *answer* as:
11 : left if only the first execution failed,
12 : right if only the second execution failed,
13 : both if both executions failed.
14 : **else**
15 : $\sigma_{0\oplus b} \leftarrow \mathsf{Open}(\tau_{0\oplus b}, s_0)$
16 : $\sigma_{1\oplus b} \leftarrow \mathsf{Open}(\tau_{1\oplus b}, s_1)$
17 : $answer = (\sigma_0, \sigma_1)$
18 : $b' \leftarrow \mathcal{A}(answer)$
19 : **return** $(b = b')$

informed choice of which execution to abort, we can ignore this scenario and hence let the experiment abort in Line 6.

7 Blind Conditional Signatures Construction

In this chapter, we will give our construction of a provably secure blind conditional signature scheme in the random oracle model. Our construction is a variation of the original A2L+ protocol by Glaeser et al. [GMM+22]. The diverging parts are marked. The **send** and **receive** statements indicate secure and private communication between the parties. It is assumed that the public parameters are available to all participants. Furthermore, the encryption key ek of the Hub, as well as all verification keys vk, are publicly known. In case the blind conditional signature scheme is used as a coin-mixing protocol, we further assume that the underlying blockchain supports time locks. In this scenario, the messages m_{AH} and m_{HB} act as transaction messages. To ensure that the coins contained in these transaction messages are not spent before the other party completes the pre-signatures, we assume these coins are locked for the duration of the protocol execution. To solve this, Glaeser et al. [GMM+22] require a payment channel setup before the protocol starts. We leave this open to choose the preferred method of implementation, as payment channels may not be available on all blockchains with time-locking capabilities. Moreover, using a constant amount of coins for each transaction is necessary as it otherwise becomes trivial to link two payments and break blindness. Similar to the original A2L+ protocol, we further assume that the protocol runs in epochs, where each epoch comprises three phases corresponding to the $\mathsf{PPromise}$, $\mathsf{PSolver}$, and Open executions, respectively. To protect against griefing attacks for coin-mixing protocols, the registration protocol described in Section 3.3.3 must be executed beforehand.

In Construction 3, we give the construction for our revised A2L+ blind conditional signature protocol and claim the security thereof in Theorem 1. We defer the formal security proofs to Chapter 8.

Construction 3 (A2L+ Blind Conditional Signature Scheme)**.** *We assume the existence of group description parameter* $\mathsf{gp} = (\mathbb{G}, g, q)$*. Let* Rel *be a canonical hard relation with* PPT *sampling algorithm* $\mathsf{GenR}(1^\lambda)$*. Let* $\Pi_{RP} = (\mathsf{PSetup}, \mathsf{PGen}, \mathsf{PSolve}, \mathsf{PRand})$ *be a secure randomizable puzzle scheme,* $\mathsf{NIZK} = (\mathsf{Setup}, \mathsf{P}, \mathsf{V}, \mathsf{Rand})$ *be a sound randomizable non-interactive zero-knowledge proof system for the language* $\mathcal{L}_{\mathsf{NIZK}} := \{(\mathsf{ek}, Y, c) | \exists y \in \mathbb{Z}_p : g^y = Y \wedge c = \Pi_E.\mathsf{Enc}(\mathsf{ek}, y)\}$*, and* $\Pi_{AS} = (\mathsf{pSign}, \mathsf{Adapt}, \mathsf{pVrf}, \mathsf{Extract})$ *be a secure adaptor signature scheme. We define the blind conditional signature scheme* $A2L+ = (\mathsf{PPromise}, \mathsf{PSolver}, \mathsf{Open})$ *in Figure 7.1, Figure 7.2, and Figure 7.3 respectively.*

Theorem 1. *We assume the existence of group description parameter* $\mathsf{gp} = (\mathbb{G}, g, q)$. *Let* Rel *be a canonical hard relation with* PPT *sampling algorithm* $\mathsf{GenR}(1^\lambda)$. *Let* Π_{RP} *be a secure randomizable puzzle scheme according to Construction 1 that leverages an* IND-CCA*-secure LOE scheme* Π_E. *Let* NIZK *be a sound randomizable non-interactive zero-knowledge proof system, and let* Π_{AS} *be a secure adaptor signature scheme that achieves pre-signature correctness, extractability, unique extractability, unlinkability, pre-verify soundness, and pre-signature adaptability. Assuming the hardness of the OMDL, the A2L+ protocol is a secure blind conditional signature scheme.*

7.1 Puzzle Promise Protocol

In the puzzle promise protocol PPromise, we add a randomized NIZK π' to the output. After verifying the NIZK π and pre-signature $\widetilde{\sigma}$ that the Hub sent, the randomizable puzzle returns a randomized version of the puzzle Z' alongside the randomization factor r. We use r to run the Rand algorithm on the NIZK π and receive a randomized version π', which we add at the end of the output τ. π' proofs that the randomized ciphertext c' contained in the puzzle Z' encrypts a valid secret y' to the randomized statement Y' under the encryption key of the Hub ek^H. The PPromise protocol is formally described in Figure 7.1.

Figure 7.1: Puzzle promise protocol.

Public Parameters: group description $\mathsf{pp} := (\mathbb{G}, g, q)$, message m_{HB}

$\mathsf{PPromise}\langle H(\tilde{\mathsf{dk}}, \mathsf{sk}^H, m_{HB}), \cdot \rangle$

1: $(Y, y) \leftarrow \mathsf{Rel.GenR}(1^\lambda)$
2: $Z \leftarrow \Pi_{\mathsf{RP}}.\mathsf{PGen}((\mathsf{pp}, \tilde{\mathsf{ek}}), y)$
3: **parse** $(Y, c) := Z$
4: $\pi \leftarrow \mathsf{NIZK.P}((\tilde{\mathsf{ek}}, Y, c), y)$
5: $\widetilde{\sigma}_{HB} \leftarrow \Pi_{\mathsf{AS}}.\mathsf{pSign}(\mathsf{sk}^H, m_{HB}, Y)$
6: **send** $(Z, \pi, \widetilde{\sigma}_{HB})$
7: **return** $\perp$

$\mathsf{PPromise}\langle \cdot, B(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB}) \rangle$

1: **receive** $(Z, \pi, \widetilde{\sigma}_{HB})$
2: **if** $\mathsf{NIZK.V}((\tilde{\mathsf{ek}}, Z), \pi) \neq 1$ **then**
3: **return** $\perp$
4: **parse** $(Y, c) := Z$
5: **if** $\Pi_{\mathsf{AS}}.\mathsf{pVrf}(\mathsf{vk}^H, m_{HB}, Y, \widetilde{\sigma}_{HB}) \neq 1$
6: **then return** $\perp$
7: $(Z', r) \leftarrow \Pi_{\mathsf{RP}}.\mathsf{PRand}((\mathsf{pp}, \tilde{\mathsf{ek}}), Z)$
8: $\pi' \leftarrow \mathsf{NIZK.Rand}((\tilde{\mathsf{ek}}, Z), \pi, r)$
9: $\tau := (r, m_{HB}, \widetilde{\sigma}_{HB}, Z', \pi')$
10: **return** τ

7.2 Puzzle Solver Protocol

In the puzzle solver protocol PSolver, Alice additionally checks if the randomized NIZK π' is valid and aborts the execution by returning $\perp$ in case it is not. As we pointed out in Section 5.1.2, this additional check protects Alice against collusion between the Hub and Bob. Apart from that, we argue that Alice should be able to prove in some way the validity of the puzzle she receives before initiating the puzzle solver protocol since she also locks her coins in the pre-signature $\tilde{\sigma}_{AH}$ based on the statement Y''. The randomized NIZK π' allows Alice to verify that the Hub can solve the randomized puzzle Z' only by using his decryption key $\tilde{\mathsf{dk}}$ as the trapdoor. This holds since π' proofs that the ciphertext c' encrypts a valid solution to the statement Y' contained in Z' under the public encryption key $\tilde{\mathsf{ek}}$ of the Hub. As already mentioned in Section 5.1.2, our Construction 1 of a randomizable puzzle scheme requires the public encryption key $\tilde{\mathsf{ek}}$ used to generate the original puzzle Z to randomize the puzzle in the PRand algorithm. The randomized puzzle Z'' otherwise can not be solved by using the decryption key of the Hub $\tilde{\mathsf{dk}}$ as the trapdoor. Hence, when using such a Π_{RP} scheme and the $\tilde{\mathsf{ek}}$ that Alice uses in the PRand algorithm is different from the one used by the Hub to generate the original puzzle Z, the Hub can not solve the puzzle Z'' and the PSolver execution will fail. Since not all randomizable puzzle schemes satisfy this inherent property and to protect Alice from unnecessarily locking her coins, we explicitly add the randomized NIZK π' into our construction. The PSolver protocol is formally described in Figure 7.2.

Figure 7.2: Puzzle solver protocol.

Public Parameters: group description $\mathsf{pp} := (\mathbb{G}, g, q)$, message m_{AH}

$\mathsf{PSolver}\langle A(\mathsf{sk}^A, \tilde{\mathsf{ek}}, m_{AH}, \tau), \cdot \rangle$

1: **parse** $\tau := (\cdot, \cdot, \cdot, Z', \pi')$
2: **if** $\mathsf{NIZK.V}((\tilde{\mathsf{ek}}, Z'), \pi') \neq 1$ **then**
3: **return** $\perp$
4: $(Z'', r') \leftarrow \Pi_{\mathsf{RP}}.\mathsf{PRand}((\mathsf{pp}, \tilde{\mathsf{ek}}), Z')$
5: **parse** $(Y'', c'') := Z''$
6: $\tilde{\sigma}_{AH} \leftarrow \Pi_{\mathsf{AS}}.\mathsf{pSign}(\mathsf{sk}^A, m_{AH}, Y'')$
7: **send** $(Z'', \tilde{\sigma}_{AH})$
8: **receive** σ_{AH}
9: $y'' \leftarrow \Pi_{\mathsf{AS}}.\mathsf{Extract}(\mathsf{vk}^A, \tilde{\sigma}_{AH}, \sigma_{AH}, Y'')$
10: **if** $y'' = \perp$ **then return** $\perp$
11: $y' := y'' - r'$
12: **return** (σ_{AH}, y')

$\mathsf{PSolver}\langle \cdot, H(\tilde{\mathsf{dk}}, \mathsf{vk}^A, m_{AH}) \rangle$

1: **receive** $(Z'', \tilde{\sigma}_{AH})$
2: $y'' \leftarrow \Pi_{\mathsf{RP}}.\mathsf{PSolve}(\tilde{\mathsf{dk}}, Z'')$
3: **if** $\Pi_{\mathsf{AS}}.\mathsf{pVrf}(\mathsf{vk}^A, m_{AH}, Y'', \tilde{\sigma}_{AH}) \neq 1$
4: $\vee\ g^{y''} \neq Y''$ **then return** $\perp$
5: $\sigma_{AH} \leftarrow \Pi_{\mathsf{AS}}.\mathsf{Adapt}(\mathsf{vk}^A, \tilde{\sigma}_{AH}, y'')$
6: **send** σ_{AH}
7: **return** σ_{AH}

7.3 Open Algorithm

The Open algorithm stays unchanged. We restate it in Figure 7.3 for completeness. On input the puzzle τ and the witness y' output by Alice after successful completion of the $\mathsf{PSolver}$ protocol, we derandomize y' to a valid solution y of the original statement Y embedded in the pre-signature $\widetilde{\sigma}_{HB}$. Finally, the Adapt algorithm is executed to receive a valid full signature on the message m_{HB}.

Figure 7.3: Open algorithm.

$\mathsf{Open}(\tau, y')$

1 : **parse** $\tau := (r, \cdot, \widetilde{\sigma}, \cdot, \cdot)$
2 : $y := y' - r$
3 : $\sigma := \Pi_{\mathsf{AS}}.\mathsf{Adapt}(\mathsf{vk}^H, \widetilde{\sigma}, y)$
4 : **return** σ

8 Security Analysis

In this chapter, we conduct the security analysis of blind conditional signatures by proofing Theorem 1. We argue about each property separately, using game-based proofs.

8.1 Selective-Failure Blindness

Lemma 1 (Selective-Failure Blindness)**.** *We assume the existence of group description parameter* $\mathsf{gp} = (\mathbb{G}, g, q)$*. Let* Rel *be a canonical hard relation with* PPT *sampling algorithm* $\mathsf{GenR}(1^\lambda)$*. Let* Π_{RP} *be a secure randomizable puzzle scheme according to Construction 1 that leverages an* IND-CCA*-secure LOE scheme* Π_{E}*. Let* NIZK *be a sound randomizable non-interactive zero-knowledge proof system, and let* Π_{AS} *be a secure adaptor signature scheme that achieves pre-signature correctness, extractability, unique extractability, unlinkability, pre-verify soundness, and pre-signature adaptability. Assuming the hardness of the OMDL, the A2L+ protocol satisfies selective-failure blindness (c.f. Definition 25) in the LOE model.*

Proof of Lemma 1. We prove Lemma 1 by contradiction, assuming that there exists a PPT adversary $\mathcal{A}$ that wins the $\mathsf{ExpSFBlnd}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ experiment with non-negligible probability. In the following, we will perform a series of game hops with negligible transitions, starting from the original game $\mathsf{ExpSFBlnd}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$. Eventually, we will demonstrate that under the stated assumptions, no PPT adversary can succeed with probability significantly larger than the random choice in the final game, thus contradicting the initial assumption.

Game $\mathcal{G}_0$: The first game $\mathcal{G}_0$ is the original $\mathsf{ExpSFBlnd}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ game in Figure 4.1. Since $\mathcal{G}_0$ simulates the game $\mathsf{ExpSFBlnd}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ perfectly, it holds that $\Pr\left[\mathsf{ExpSFBlnd}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}} = 1\right] = \Pr[\mathcal{G}_0(\lambda) = 1]$. A formal description of the game $\mathcal{G}_0$ is given in Figure 8.1.

Game $\mathcal{G}_1$: In the second game, we exchange the executions of the Open algorithm with an oracle $\mathcal{O}\mathsf{Open}$ that always returns the adapted signature to the respective pre-signature contained in the puzzle τ. The $\mathcal{O}\mathsf{Open}$ oracle generates all statement/witness pairs in the puzzle promise $\mathsf{PPromise}$ protocol and therefore knows all witnesses to the embedded statements. A formal description of the game $\mathcal{G}_1$ is given in Figure 8.2.

Claim 1. *$\mathcal{G}_1$ can only be distinguished from $\mathcal{G}_0$ if the secrets used to call the Open algorithms differ from the witnesses used during the puzzle creation in the $\mathsf{PSolver}$*

Figure 8.1: The first blindness game $\mathcal{G}_0(\lambda)$.

$\mathcal{G}_0(\lambda)$

1 : $(\tilde{\mathsf{ek}}, \mathsf{vk}_0^H, \mathsf{vk}_1^H, (m_{HB,0}, m_{AH,0}), (m_{HB,1}, m_{AH,1})) \leftarrow \mathcal{A}(1^\lambda)$
2 : $(\mathsf{vk}_0^A, \mathsf{sk}_0^A) \leftarrow \mathsf{KGen}(1^\lambda)$
3 : $(\mathsf{vk}_1^A, \mathsf{sk}_1^A) \leftarrow \mathsf{KGen}(1^\lambda)$
4 : $\tau_0 \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}_0^A, \mathsf{vk}_1^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}_0^H, m_{HB,0}) \Big\rangle$
5 : $\tau_1 \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}_0^A, \mathsf{vk}_1^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}_1^H, m_{HB,1}) \Big\rangle$
6 : **if** $(\tau_0 = \perp) \vee (\tau_1 = \perp)$ **then return** 0
7 : $b \leftarrow \{0, 1\}$
8 : $(\sigma_0^*, s_0) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}_0^A, \tilde{\mathsf{ek}}, m_{AH,0}, \tau_{0\oplus b}), \mathcal{A} \Big\rangle$
9 : $(\sigma_1^*, s_1) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}_1^A, \tilde{\mathsf{ek}}, m_{AH,1}, \tau_{1\oplus b}), \mathcal{A} \Big\rangle$
10 : **if** $(\sigma_0^* = \perp) \vee (\sigma_1^* = \perp)$ define *answer* as:
11 : left if only the first execution failed,
12 : right if only the second execution failed,
13 : both if both executions failed.
14 : **else**
15 : $\sigma_{0\oplus b} \leftarrow \mathsf{Open}(\tau_{0\oplus b}, s_0)$
16 : $\sigma_{1\oplus b} \leftarrow \mathsf{Open}(\tau_{1\oplus b}, s_1)$
17 : $answer = (\sigma_0, \sigma_1)$
18 : $b' \leftarrow \mathcal{A}(answer)$
19 : **return** $(b = b')$

execution. Let $\mathsf{Break}_{\mathsf{UniqueExtractability}}$ *be the event where the secrets output by the* PSolver *algorithm differ from the witnesses embedded in the ciphertext of the puzzle* τ*. Then, the adversary* $\mathcal{A}$ *returned a valid signature* σ_i^* *that extracts to more than one witness. If* $\mathsf{Break}_{\mathsf{UniqueExtractability}}$ *happens, we can build a reduction that breaks unique extractability of the underlying adaptor signature scheme* Π_{AS}*. Therefore,* $\mathsf{Break}_{\mathsf{UniqueExtractability}}$ *can occur only with negligible probability. Since* $\Pr[\mathcal{G}_1 \wedge \neg \mathsf{Break}_{\mathsf{UniqueExtractability}}] = \Pr[\mathcal{G}_0 \wedge \neg \mathsf{Break}_{\mathsf{UniqueExtractability}}]$ *and the event* $\mathsf{Break}_{\mathsf{UniqueExtractability}}$ *occurring is negligible,* $\mathcal{G}_0$ *and* $\mathcal{G}_1$ *need to be indistinguishable. It holds that* $\Pr[\mathcal{G}_1] \approx \Pr[\mathcal{G}_0]$.

Proof of Claim 1. From Line 10 in the PSolver protocol, it follows that for any successful execution of the PSolver protocol, the secrets s_0, s_1 are valid witnesses to the embedded statements. The embedded randomized statements used in the puzzle τ, in turn, have been generated by calculating a group multiplication on the original statement $Y' = Y \cdot g^r$. Therefore, each valid witness to Y' can be derandomized to a valid witness of Y with the help of the randomization factor $y = y' - r$, which is part of

Figure 8.2: The second blindness game $\mathcal{G}_1(\lambda)$. The $\mathcal{O}\mathsf{Open}$ oracle generates all statement/witness pairs during the $\mathsf{PPromise}$ protocol and therefore knows all witnesses to statements generated during $\mathsf{PPromise}$ executions.

$\mathcal{G}_1(\lambda)$

1 : $(\tilde{\mathsf{ek}}, \mathsf{vk}_0^H, \mathsf{vk}_1^H, (m_{HB,0}, m_{AH,0}), (m_{HB,1}, m_{AH,1})) \leftarrow \mathcal{A}(1^\lambda)$
2 : $(\mathsf{vk}_0^A, \mathsf{sk}_0^A) \leftarrow \mathsf{KGen}(1^\lambda)$
3 : $(\mathsf{vk}_1^A, \mathsf{sk}_1^A) \leftarrow \mathsf{KGen}(1^\lambda)$
4 : $\tau_0 \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}_0^A, \mathsf{vk}_1^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}_0^H, m_{HB,0}) \Big\rangle$
5 : $\tau_1 \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}_0^A, \mathsf{vk}_1^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}_1^H, m_{HB,1}) \Big\rangle$
6 : **if** $(\tau_0 = \perp) \vee (\tau_1 = \perp)$ **then return** 0
7 : $b \leftarrow \{0, 1\}$
8 : $(\sigma_0^*, s_0) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}_0^A, \tilde{\mathsf{ek}}, m_{AH,0}, \tau_{0\oplus b}), \mathcal{A} \Big\rangle$
9 : $(\sigma_1^*, s_1) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}_1^A, \tilde{\mathsf{ek}}, m_{AH,1}, \tau_{1\oplus b}), \mathcal{A} \Big\rangle$
10 : **if** $(\sigma_0^* = \perp) \vee (\sigma_1^* = \perp)$ define *answer* as:
11 : left if only the first execution failed,
12 : right if only the second execution failed,
13 : both if both executions failed.
14 : **else**
15 : $\sigma_{0\oplus b} \leftarrow \mathcal{O}\mathsf{Open}(\tau_{0\oplus b})$
16 : $\sigma_{1\oplus b} \leftarrow \mathcal{O}\mathsf{Open}(\tau_{1\oplus b})$
17 : $answer = (\sigma_0, \sigma_1)$
18 : $b' \leftarrow \mathcal{A}(answer)$
19 : **return** $(b = b')$

$\mathcal{O}\mathsf{Open}(\tau)$

1 : **parse** $\tau := (r, m, \cdot, (Y', \cdot), \cdot); Y' := g^{y+r}$
2 : $\sigma \leftarrow \Pi_{\mathsf{AS}}.\mathsf{Open}(\mathsf{vk}, m, y)$
3 : **return** σ

the puzzle τ. The game only enters Lines 15 to 17 if both signatures σ_0^*, σ_1^* are valid, meaning the $\mathsf{PSolver}$ protocol was completed successfully. According to Construction 3, the $\mathsf{PSolver}$ algorithm only completes successfully if the two secrets s_0, s_1 are valid witnesses to the embedded statements. Therefore, the secrets used to call the Open algorithm must also be valid witnesses. It follows that from the view of $\mathcal{A}$, $\mathcal{G}_0$ and $\mathcal{G}_1$ can only be distinguished if $\mathcal{A}$ returns a valid signature σ^* in the $\mathsf{PSolver}$ protocol

that extracts to a valid witness, which is different from the witness used by the $\mathcal{O}\mathsf{Open}$ oracle and therefore different from the witness used during puzzle creation in the $\mathsf{PPromise}$ executions.

We now give a reduction that uses $\mathcal{A}$ from game $\mathcal{G}_1$ to break unique extractability of the underlying adaptor signature scheme Π_{AS}. More specifically, a reduction that uses the $\mathsf{PSolver}$ execution in $\mathcal{G}_1$. The reduction receives as input a set of verification keys vk_i and a pSign oracle. It forwards vk_i and provides an $\mathcal{O}\mathsf{PS}$ oracle to $\mathcal{A}$. On each request from $\mathcal{A}$ to the $\mathcal{O}\mathsf{PS}$ oracle, the reduction samples a statement/witness pair (Y, y) and queries the pSign oracle to receive a valid pre-signature $\tilde{\sigma}$ under a verification key vk_i. Additionally, the $\mathcal{O}\mathsf{PS}$ oracle returns the witness y encrypted under the encryption key $\tilde{\mathsf{ek}}$ of the adversary $\mathcal{A}$. The reduction now start $\mathcal{A}$ on input vk_i. First, $\mathcal{A}$ runs the $\mathsf{PPromise}$ protocol honestly, outputting a valid puzzle τ. Then, during the $\mathsf{PSolver}$ protocol $\mathcal{A}$ queries the $\mathcal{O}\mathsf{PS}$ oracle arbitrarily many times on messages m_i of its choice to receive valid pre-signatures $\tilde{\sigma}_i$ on these messages m_i alongside the encrypted witnesses c_i. Since we assume in Claim 1 that the event $\mathsf{Break}_{\mathsf{UniqueExtractability}}$ occurs with non-negligible probability, $\mathcal{A}$ eventually outputs a valid full signature σ' as the output of the $\mathsf{PSolver}$ protocol for at least one of the pre-signatures $\tilde{\sigma}_i$ that extracts to a valid witness y' that is different from the witness y in the ciphertext c from the $\mathcal{O}\mathsf{PS}$ oracle. Furthermore, the adversary $\mathcal{A}$ decrypts the ciphertext c using its decryption key $\tilde{\mathsf{dk}}$ to receive the original witness y and runs the adapt algorithm $\Pi_{\mathsf{AS}}.\mathsf{Adapt}$ to receive a second full signature σ on the same pre-signature $\tilde{\sigma}$. The reduction finally outputs the two signatures (σ', σ) to break unique extractability as in Definition 15 of the adaptor signature scheme Π_{AS}, since the two signatures (σ', σ) are valid, distinct, and extract to different witnesses using the same valid pre-signature $\tilde{\sigma}$. Furthermore, by assumption, $\mathcal{A}$ runs in polynomial time, and the algorithms used to simulate the $\mathcal{O}\mathsf{PS}$ oracle are all PPT algorithms. Thus, the reduction is efficient and contradicts unique extractability of the adaptor signature scheme Π_{AS}. Hence, the event $\mathsf{Break}_{\mathsf{UniqueExtractability}}$ can only occur with negligible probability. It holds, that $\mathcal{G}_0$ and $\mathcal{G}_1$ are indistinguishable. □

Game $\mathcal{G}_2$: In the third game, we replace the randomization of the randomizable puzzle with a freshly sampled instance. A formal description of the game $\mathcal{G}_2$ is given in Figure 8.3.

Claim 2. *From the view of the adversary $\mathcal{A}$, the randomized version of the puzzle that the $\mathsf{PSolver}$ algorithm receives as input looks like a freshly sampled puzzle. Therefore, randomizing a valid puzzle of the Π_{RP} scheme is the same as sampling a new instance from the view of $\mathcal{A}$, and any distinguishing advantage necessarily corresponds to the case in which the adversary is given an invalid puzzle during the $\mathsf{PSolver}$ execution. Let $\mathsf{Break}_{\mathsf{Soundness}}$ be the event where the adversary receives an invalid puzzle in the $\mathsf{PSolver}$ execution. Then, we can build a reduction that breaks the soundness of the NIZK proof system. Hence, $\mathsf{Break}_{\mathsf{Soundness}}$ can only happen with negligible probability. Since $\Pr[\mathcal{G}_2 \wedge \neg \mathsf{Break}_{\mathsf{Soundness}}] = \Pr[\mathcal{G}_1 \wedge \neg \mathsf{Break}_{\mathsf{Soundness}}]$ the two games $\mathcal{G}_1$ and $\mathcal{G}_2$ need to be indistinguishable. It holds that $\Pr[G_1] \approx \Pr[\mathcal{G}_2]$.*

Figure 8.3: The third blindness game $\mathcal{G}_2(\lambda)$. The $\mathcal{O}\mathsf{Open}$ oracle generates all statement/witness pairs during the $\mathsf{PPromise}$ protocol and therefore knows all witnesses to statements generated during $\mathsf{PPromise}$ executions.

$\mathcal{G}_2(\lambda)$

$(\tilde{\mathsf{ek}}, \mathsf{vk}_0^H, \mathsf{vk}_1^H, (m_{HB,0}, m_{AH,0}), (m_{HB,1}, m_{AH,1})) \leftarrow \mathcal{A}(1^\lambda)$
$(\mathsf{vk}_0^A, \mathsf{sk}_0^A) \leftarrow \mathsf{KGen}(1^\lambda)$
$(\mathsf{vk}_1^A, \mathsf{sk}_1^A) \leftarrow \mathsf{KGen}(1^\lambda)$
$\tau_0 \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}_0^A, \mathsf{vk}_1^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}_0^H, m_{HB,0}) \Big\rangle$
$\tau_1 \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}_0^A, \mathsf{vk}_1^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}_1^H, m_{HB,1}) \Big\rangle$
if $(\tau_0 = \perp) \vee (\tau_1 = \perp)$ **then return** 0
$(Y_0', s_0), (Y_1', s_1) \leftarrow \mathsf{Rel.GenR}(1^\lambda)$
$Z_0' \leftarrow \Pi_{\mathsf{RP}}.\mathsf{PGen}((\mathsf{pp}, \tilde{\mathsf{ek}}), s_0)$
$Z_1' \leftarrow \Pi_{\mathsf{RP}}.\mathsf{PGen}((\mathsf{pp}, \tilde{\mathsf{ek}}), s_1)$
$\pi_0' \leftarrow \mathsf{NIZK.P}((\tilde{\mathsf{ek}}, Z_0'), s_0)$
$\pi_1' \leftarrow \mathsf{NIZK.P}((\tilde{\mathsf{ek}}, Z_1'), s_1)$
$\tau_0' := (\cdot, \cdot, \cdot, \cdot, Z_0', \pi_0')$
$\tau_1' := (\cdot, \cdot, \cdot, \cdot, Z_1', \pi_1')$
$b \leftarrow \{0, 1\}$
$(\sigma_0^*, s_0) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}_0^A, \tilde{\mathsf{ek}}, m_{AH,0}, \tau'_{0\oplus b}), \mathcal{A} \Big\rangle$
$(\sigma_1^*, s_1) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}_1^A, \tilde{\mathsf{ek}}, m_{AH,1}, \tau'_{1\oplus b}), \mathcal{A} \Big\rangle$
if $(\sigma_0^* = \perp) \vee (\sigma_1^* = \perp)$ define *answer* as:
  left if only the first execution failed,
  right if only the second execution failed,
  both if both executions failed.
else
  $\sigma_{0\oplus b} \leftarrow \mathcal{O}\mathsf{Open}(\tau_{0\oplus b})$
  $\sigma_{1\oplus b} \leftarrow \mathcal{O}\mathsf{Open}(\tau_{1\oplus b})$
  $answer = (\sigma_0, \sigma_1)$
$b' \leftarrow \mathcal{A}(answer)$
return $(b = b')$

$\mathcal{O}\mathsf{Open}(\tau)$

parse $\tau := (r, m, \cdot, (Y', \cdot), \cdot); Y' := g^{y+r}$
$\sigma \leftarrow \Pi_{\mathsf{AS}}.\mathsf{Open}(\mathsf{vk}, m, y)$
return σ

Proof of Claim 2. In the puzzle solver protocol $\mathsf{PSolver}$ in Line 4, the puzzle gets rerandomized using the PRand algorithm of the randomizable puzzle scheme Π_{RP} before it gets sent to the adversary $\mathcal{A}$. According to Construction 1 this is done by a group multiplication on the statement $Y'' = Y' \cdot g^{r'}$ of the hard relation that is embedded in τ and by multiplying the encryption of the witness with the encryption of the uniform element under the encryption key of the Hub $c'' = c' \cdot \Pi_{\mathsf{E}}.\mathsf{Enc}(\tilde{\mathsf{ek}}, r')$. Since r' is unknown to $\mathcal{A}$, the resulting rerandomized puzzle Z'' therefore looks similar to a freshly sampled puzzle given that the original ciphertext c from the adversary $\mathcal{A}$ is well-formed and decrypts to a valid witness for the embedded statement Y''. Alice verifies the validity of the puzzle she receives from Bob by verifying an NIZK during the $\mathsf{PSolver}$ execution before rerandomizing the puzzle. Hence, any distinguishing advantage necessarily implies a violation of the soundness property of the NIZK. We now give a reduction that uses $\mathcal{A}$ from $\mathcal{G}_2$ to break soundness of the NIZK proof system. The language of the NIZK proof system used in the A2L+ protocol is defined as $\mathcal{L}_{\mathsf{NIZK}} := \{(\mathsf{ek}, Y, c) | \exists y \in \mathbb{Z}_p : g^y = Y \wedge c = \Pi_{\mathsf{E}}.\mathsf{Enc}(\mathsf{ek}, y)\}$. The reduction generates a set of key pairs $(\mathsf{vk}_i, \mathsf{sk}_i)$ and runs $\mathcal{A}$ in input vk_i. The adversary $\mathcal{A}$ first generates a key pair $(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}})$. Then $\mathcal{A}$ chooses a random statement witness pair (Y, y) during the $\mathsf{PPromise}$ protocol, encrypts y in the ciphertext c under its encryption key $\tilde{\mathsf{ek}}$, such that $g^{\Pi_{\mathsf{E}}.\mathsf{Dec}(\tilde{\mathsf{dk}}, c)} \neq Y$, and creates a proof $\pi \leftarrow \mathsf{NIZK.P}((\tilde{\mathsf{ek}}, Y, c), y)$. The reduction finally outputs the proof π. Since we assume in Claim 2 that $\mathsf{Break}_{\mathsf{Soundness}}$ occurs and Alice accepts an invalid puzzle in the $\mathsf{PSolver}$ protocol, π is a valid proof with non-negligible probability, such that $\mathsf{NIZK.V}((\tilde{\mathsf{ek}}, Y, c), \pi) = 1$. By assumption, $\mathcal{A}$ is a PPT algorithm, and the reduction is efficient. This reduction violates the soundness property of the NIZK proof system because $(\tilde{\mathsf{ek}}, Y, c) \notin \mathcal{L}_{\mathsf{NIZK}}$. Thus, $\mathsf{Break}_{\mathsf{Soundness}}$ can only happen with negligible probability, and it holds that $\Pr[G_1] \approx \Pr[\mathcal{G}_2]$. □

Claim 3. *There exists no polynomially bound adversary that wins game $\mathcal{G}_2$ with probability significantly larger than $1/2$, and it holds that*

$$\Pr[G_2 = 1] \leq 1/2 + \mathsf{negl}(\lambda).$$

Proof of Claim 3. During the $\mathsf{PSolver}$ executions in $\mathcal{G}_2$, the adversary is given valid pre-signatures on the messages $m_{AH,0}$ and $m_{AH,1}$ of its choice signed under the signing keys $\mathsf{sk}_0^A, \mathsf{sk}_1^A$, alongside freshly sampled puzzles. Since the two puzzles have been generated by sampling a witness uniformly at random, the adversary has no way of distinguishing which bit b has been used with probability significantly larger than the random choice. Aborting any of the two $\mathsf{PSolver}$ executions will not return any further information. In case the protocol executes successfully, the adversary is given valid full signatures on the pre-signatures contained in τ_0, τ_1 of the $\mathsf{PPromise}$ protocol. These full signatures are independent of the puzzles used in the $\mathsf{PSolver}$ protocol. Hence, the adversary can not learn any distinguishing information in G_2, and the two cases of $b = 0$ and $b = 1$ are indistinguishable. □

We can now use Claim 3 to complete our proof of Lemma 1. Through a series of game hops, we showed that under the stated assumptions, the probability of winning the

blindness experiment $\mathsf{ExpSFBlind}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ is indistinguishable to the probability of winning the final game $\mathcal{G}_2$. In Claim 3 we furthermore proved, that the two cases $b = 0$ and $b = 1$ are indistinguishable, and it holds that

$$\Pr\left[\mathsf{ExpSFBlind}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}} = 1\right] \leq \Pr[\mathcal{G}_2 = 1] + \mathsf{negl}(\lambda) \leq 1/2 + \mathsf{negl}(\lambda).$$

From the assumption that there exists a PPT adversary $\mathcal{A}$ that wins the original blindness experiment $\mathsf{ExpSFBlind}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ with probability significantly larger than $1/2$, it follows that $\mathcal{A}$ also wins $\mathcal{G}_2$ with probability significantly larger than $1/2$. This contradicts Claim 3, and thus, no such efficient adversary can exist, and A2L+ satisfies selective-failure blindness.

□

8.2 Unlockability

Lemma 2 (Unlockability)**.** *We assume the existence of group description parameter* $gp = (\mathbb{G}, g, q)$*. Let* Rel *be a canonical hard relation with* PPT *sampling algorithm* $\mathsf{GenR}(1^\lambda)$*. Let* Π_{RP} *be a secure randomizable puzzle scheme according to Construction 1 that leverages an* IND-CCA*-secure LOE scheme* Π_E*. Let* NIZK *be a sound randomizable non-interactive zero-knowledge proof system, and let* Π_{AS} *be a secure adaptor signature scheme that achieves pre-signature correctness, extractability, unique extractability, unlinkability, pre-verify soundness, and pre-signature adaptability. Assuming the hardness of the OMDL, the A2L+ protocol satisfies unlockability (c.f. Definition 22) in the LOE model.*

Proof of Lemma 2. We prove this lemma by showing through a series of game hops with negligible transitions that, given the stated assumptions, there can not exist a PPT adversary winning the $\mathsf{ExpUnlock}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ game with probability significantly larger than negligible. We do this by introducing several statements under which the game aborts and showing that the probability of each event happening must be negligible. Finally, we conclude the proof by contradiction, as no PPT adversary can exist with non-negligible success probability in the final game.

Game $\mathcal{G}_0$: The first game $\mathcal{G}_0$ is the original $\mathsf{ExpUnlock}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ game in Figure 4.2. $\mathcal{G}_0$ simulates $\mathsf{ExpUnlock}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ perfectly, and it therefore holds that $\Pr\left[\mathsf{ExpUnlock}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}} = 1\right] = \Pr[\mathcal{G}_0 = 1]$. The formal description of $\mathcal{G}_0$ is given in Figure 8.4.

Game $\mathcal{G}_1$: The second game $\mathcal{G}_1$ differs from game $\mathcal{G}_0$ when it aborts in Line 6 instead of returning the winning condition. This event only occurs when the adversary outputs a valid signature $\hat{\sigma}$ under Alice's verification key vk^A while the $\mathsf{PPromise}$ algorithm returns $\perp$, effectively returning a valid forgery. This game hop does not differ from $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ if this event does not occur. The formal description of $\mathcal{G}_1$ is given in Figure 8.5.

Figure 8.4: The first unlockability game $\mathcal{G}_0(\lambda)$.

$\mathcal{G}_0(\lambda)$

1 : $(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB}, m_{AH}) \leftarrow \mathcal{A}(1^\lambda)$
2 : $(\mathsf{vk}^A, \mathsf{sk}^A) \leftarrow \mathsf{KGen}(1^\lambda)$
3 : $\tau \leftarrow \mathsf{PPromise}\langle \mathcal{A}(\mathsf{vk}^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB})\rangle$
4 : **if** $\tau = \perp$
5 : $\quad(\hat{\sigma}, \hat{m}) \leftarrow \mathcal{A}$
6 : $\quad b_0 := (\mathsf{Vf}(\mathsf{vk}^A, \hat{\sigma}, \hat{m}) = 1)$
7 : **if** $\tau \neq \perp$
8 : $\quad(\sigma^*, s) \leftarrow \mathsf{PSolver}\langle A(\mathsf{sk}^A, \tilde{\mathsf{ek}}, m_{AH}, \tau), \mathcal{A}\rangle$
9 : $\quad(\hat{\sigma}, \hat{m}) \leftarrow \mathcal{A}$
10 : $\quad b_1 := (\mathsf{Vf}(\mathsf{vk}^A, \hat{\sigma}, \hat{m}) = 1) \wedge (\hat{m} \neq m_{AH}))$
11 : $\quad b_2 := (\mathsf{Vf}(\mathsf{vk}^A, \sigma^*, m_{AH}) = 1)$
12 : $\quad b_3 := (\mathsf{Vf}(\mathsf{vk}^H, m_{HB}, \mathsf{Open}(\tau, s)) \neq 1)$
13 : **return** $b_0 \vee b_1 \vee (b_2 \wedge b_3)$

Claim 4. *Let* $\mathsf{Break}_{\mathsf{Extract}}$ *be the event in Line 6, where* $\mathcal{G}1$ *aborts. Then, the adversary* $\mathcal{A}$ *found a valid signature forgery* $\hat{\sigma}$ *on any message* $\hat{m}$ *under the verification key* vk^A*. If* $\mathsf{Break}_{\mathsf{Extract}}$ *happens, then we can build a reduction that can break extractability of the underlying adaptor signature scheme* Π_{AS}*. Therefore,* $\mathsf{Break}_{\mathsf{Extract}}$ *can occur only with negligible probability. Since* $\Pr[\mathcal{G}_1 \wedge \neg \mathsf{Break}_{\mathsf{Extract}}] = \Pr[\mathcal{G}_0 \wedge \neg \mathsf{Break}_{\mathsf{Extract}}]$ *and the event* $\mathsf{Break}_{\mathsf{Extract}}$ *occurring is negligible,* $\mathcal{G}_0$ *and* $\mathcal{G}_1$ *need to be indistinguishable. It holds that* $\Pr[\mathcal{G}_1] \approx \Pr[\mathcal{G}_0]$*.*

Proof of Claim 4. We prove Claim 4 with a reduction that uses $\mathcal{A}$ from game $\mathcal{G}_1$ to break extractability of the underlying adaptor signature scheme Π_{AS}. The reduction is given as input, a verification key vk^A, and has access to a pSign oracle that, on input, a message m and statement Y, returns a pre-signature $\tilde{\sigma}$ on m under the verification key vk^A as in Definition 14. The pSign oracle is efficient, as it follows the PPT pre-signing algorithm $\Pi_{\mathsf{AS}}.\mathsf{pSign}$. The reduction forwards the pSign oracle to $\mathcal{A}$ and runs $\mathcal{A}$ on input vk^A. First $\mathcal{A}$ now generates a key pair $(\mathsf{vk}^H, \mathsf{sk}^H)$ using the $\Pi_{\mathsf{AS}}.\mathsf{KGen}$ algorithm and chooses arbitrary messages m_{AH} and m_{HB}. The adversary $\mathcal{A}$ then returns an invalid pre-signature on the message m_{HB} under its own singing key sk^H during the $\mathsf{PPromise}$ execution in Line 3 of $\mathcal{G}_1$. This leads to an invalid puzzle $\tau = \perp$ as the output of the $\mathsf{PPromise}$ protocol. Since we assume by contradiction in Claim 4 that $\mathsf{Break}_{\mathsf{Extract}}$ happens with non-negligible probability, $\mathcal{A}$ eventually outputs a valid signature $\hat{\sigma}$ on any message $\hat{m}$ under Alice's verification key vk^A without querying the pSign oracle. The reduction can then use the output $\hat{\sigma}$ to break extractability of the underlying

Figure 8.5: The second unlockability game $\mathcal{G}_1(\lambda)$.

$\mathcal{G}_1(\lambda)$

1 : $(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB}, m_{AH}) \leftarrow \mathcal{A}(1^\lambda)$
2 : $(\mathsf{vk}^A, \mathsf{sk}^A) \leftarrow \mathsf{KGen}(1^\lambda)$
3 : $\tau \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB}) \Big\rangle$
4 : **if** $\tau = \perp$
5 : $\quad(\hat{\sigma}, \hat{m}) \leftarrow \mathcal{A}$
6 : $\quad$**if** $(\mathsf{Vf}(\mathsf{vk}^A, \hat{\sigma}, \hat{m}) = 1)$ **then abort**
7 : **if** $\tau \neq \perp$
8 : $\quad(\sigma^*, s) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}^A, \tilde{\mathsf{ek}}, m_{AH}, \tau), \mathcal{A} \Big\rangle$
9 : $\quad(\hat{\sigma}, \hat{m}) \leftarrow \mathcal{A}$
10 : $\quad b_1 := (\mathsf{Vf}(\mathsf{vk}^A, \hat{\sigma}, \hat{m}) = 1) \wedge (\hat{m} \neq m_{AH}))$
11 : $\quad b_2 := (\mathsf{Vf}(\mathsf{vk}^A, \sigma^*, m_{AH}) = 1)$
12 : $\quad b_3 := (\mathsf{Vf}(\mathsf{vk}^H, m_{HB}, \mathsf{Open}(\tau, s)) \neq 1)$
13 : **return** $b_1 \vee (b_2 \wedge b_3)$

adaptor signature scheme Π_{AS} with non-negligible probability, since $\mathcal{A}$ outputs a valid signature without access to a Sign oracle (Lines 5 and 6 in Figure 2.4) and without outputting a valid pre-signature that allows extracting a valid witness to a statement $Y \in \mathsf{Rel}$ (Line 8 in Figure 2.4). Furthermore, we assume that $\mathcal{A}$ runs in polynomial time, and the reduction is therefore efficient. Thus, $\mathsf{Break}_{\mathsf{Extract}}$ can only happen with negligible probability, and the two games $\mathcal{G}_1$ and $\mathcal{G}_0$ are indistinguishable. □

Game $\mathcal{G}_2$: The game $\mathcal{G}_2$ differs from $\mathcal{G}_1$ when it aborts in Line 10 instead of returning the winning condition. This event only occurs when the adversary outputs a valid signature for a message other than m_{AH} used in the $\mathsf{PSolver}$ protocol. As in game $\mathcal{G}_1$, this can only occur if the adversary finds a valid forgery and therefore breaks extractability of Π_{AS}. If this event does not occur, $\mathcal{G}_2$ does not differ from $\mathcal{G}_1$. The formal description of $\mathcal{G}_2$ is given in Figure 8.6.

Claim 5. *Let $\mathsf{Break}_{\mathsf{Extract}}$ be the event in Line 10, where $\mathcal{G}_2$ aborts. Then the adversary $\mathcal{A}$ found a valid signature forgery $\hat{\sigma}$ under the verification key vk^A on a message $\hat{m}$ that is different from m_{AH}. If $\mathsf{Break}_{\mathsf{Extract}}$ happens, then we can build a reduction that can break extractability of the underlying adaptor signature scheme Π_{AS}. Hence, the probability of $\mathsf{Break}_{\mathsf{Extract}}$ occurring needs to be negligible. Since $\Pr[\mathcal{G}_2 \wedge \neg \mathsf{Break}_{\mathsf{Extract}}] = \Pr[\mathcal{G}_1 \wedge \neg \mathsf{Break}_{\mathsf{Extract}}]$ and the event $\mathsf{Break}_{\mathsf{Extract}}$ occurring is negligible, $\mathcal{G}_1$ and $\mathcal{G}_2$ need to be indistinguishable, and it holds that $\Pr[\mathcal{G}_2] \approx \Pr[\mathcal{G}_1]$.*

Figure 8.6: The third unlockability game $\mathcal{G}_2(\lambda)$.

$\mathcal{G}_2(\lambda)$

1 : $(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB}, m_{AH}) \leftarrow \mathcal{A}(1^\lambda)$
2 : $(\mathsf{vk}^A, \mathsf{sk}^A) \leftarrow \mathsf{KGen}(1^\lambda)$
3 : $\tau \leftarrow \mathsf{PPromise}\langle \mathcal{A}(\mathsf{vk}^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB}) \rangle$
4 : **if** $\tau = \perp$
5 : $\quad(\hat{\sigma}, \hat{m}) \leftarrow \mathcal{A}$
6 : $\quad$**if** $(\mathsf{Vf}(\mathsf{vk}^A, \hat{\sigma}, \hat{m}) = 1)$ **then abort**
7 : **if** $\tau \neq \perp$
8 : $\quad(\sigma^*, s) \leftarrow \mathsf{PSolver}\langle A(\mathsf{sk}^A, \tilde{\mathsf{ek}}, m_{AH}, \tau), \mathcal{A} \rangle$
9 : $\quad(\hat{\sigma}, \hat{m}) \leftarrow \mathcal{A}$
10 : $\quad$**if** $(\mathsf{Vf}(\mathsf{vk}^A, \hat{\sigma}, \hat{m}) = 1) \wedge (\hat{m} \neq m_{AH}))$ **then abort**
11 : $\quad b_2 := (\mathsf{Vf}(\mathsf{vk}^A, \sigma^*, m_{AH}) = 1)$
12 : $\quad b_3 := (\mathsf{Vf}(\mathsf{vk}^H, m_{HB}, \mathsf{Open}(\tau, s)) \neq 1)$
13 : **return** $(b_2 \wedge b_3)$

Proof of Claim 5. As in proof of Claim 4, we prove Claim 5 with a reduction that uses $\mathcal{A}$ from game $\mathcal{G}_2$ to break extractability of the underlying adaptor signature scheme Π_{AS}. The reduction is given as input, a verification key vk^A, and has access to a pSign oracle that, on input, a message m and statement Y, returns a pre-signature $\tilde{\sigma}$ on m under the verification key vk^A as in Definition 14. The pSign oracle is efficient, as it follows the PPT pre-signing algorithm $\Pi_{\mathsf{AS}}.\mathsf{pSign}$. The reduction forwards the pSign oracle to $\mathcal{A}$ and runs $\mathcal{A}$ on input vk^A. The adversary $\mathcal{A}$ now engages in the $\mathsf{PPromise}$ protocol to output a valid puzzle τ. Then during the $\mathsf{PSolver}$ execution in Line 8 of $\mathcal{G}_2$ the reduction parses the embedded statement from the puzzle **parse** $(Y'', c'') := Z''$ contained in τ, and queries the pSign oracle on the message m_{AH} and statement Y'' to receive a valid pre-signature $\tilde{\sigma}_{AH}$ under the verification key vk^A. The reduction sends the pre-signature $\tilde{\sigma}_{AH}$ alongside the puzzle Z'' to $\mathcal{A}$ and completes the $\mathsf{PSolver}$ protocol with $\mathcal{A}$. Since we assume in Claim 5 that $\mathsf{Break}_{\mathsf{Extract}}$ occurs with non-negligible probability, the adversary $\mathcal{A}$ eventually outputs a valid signature $\hat{\sigma}$ for a message $\hat{m}$ under Alice's verification key vk^A, such that $\hat{m} \neq m_{AH}$. Finally, the reduction uses the signature $\hat{\sigma}$ to win the extractability experiment of the adaptor signature scheme Π_{AS} as defined in Definition 14. More precisely, $\hat{\sigma}$ is a valid signature on a previously not seen message $\hat{m}$, without producing a corresponding pre-signature $\hat{\sigma}$ or an extractable witness. Since the pSign oracle and $\mathcal{A}$ run in polynomial time, the reduction is efficient and breaks extractability with non-negligible probability by the assumption of Claim 5. Thus, it holds that $\mathsf{Break}_{\mathsf{Extract}}$ can only happen with negligible probability and $\Pr[\mathcal{G}_2] \approx \Pr[\mathcal{G}_1]$. □

Figure 8.7: The fourth unlockability game $\mathcal{G}_3(\lambda)$.

$\mathcal{G}_3(\lambda)$

1 : $(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB}, m_{AH}) \leftarrow \mathcal{A}(1^\lambda)$
2 : $(\mathsf{vk}^A, \mathsf{sk}^A) \leftarrow \mathsf{KGen}(1^\lambda)$
3 : $\tau \leftarrow \mathsf{PPromise}\Big\langle \mathcal{A}(\mathsf{vk}^A), B(\tilde{\mathsf{ek}}, \mathsf{vk}^H, m_{HB}) \Big\rangle$
4 : **if** $\tau = \perp$
5 : $\quad(\hat{\sigma}, \hat{m}) \leftarrow \mathcal{A}$
6 : $\quad$**if** $(\mathsf{Vf}(\mathsf{vk}^A, \hat{\sigma}, \hat{m}) = 1)$ **then abort**
7 : **if** $\tau \neq \perp$
8 : $\quad(\sigma^*, s) \leftarrow \mathsf{PSolver}\Big\langle A(\mathsf{sk}^A, \tilde{\mathsf{ek}}, m_{AH}, \tau), \mathcal{A} \Big\rangle$
9 : $\quad(\hat{\sigma}, \hat{m}) \leftarrow \mathcal{A}$
10 : $\quad$**if** $(\mathsf{Vf}(\mathsf{vk}^A, \hat{\sigma}, \hat{m}) = 1) \wedge (\hat{m} \neq m_{AH}))$ **then abort**
11 : $\quad$**parse** $\tau := (\cdot, \cdot, \cdot, (Y', \cdot), \cdot)$
12 : $\quad$**if** $\mathsf{Vf}(\mathsf{vk}^A, \sigma^*, m_{AH}) = 1 \wedge g^s \neq Y'$ **then abort**
13 : $\quad b_2 := (\mathsf{Vf}(\mathsf{vk}^A, \sigma^*, m_{AH}) = 1)$
14 : $\quad b_3 := (\mathsf{Vf}(\mathsf{vk}^H, m_{HB}, \mathsf{Open}(\tau, s)) \neq 1)$
15 : **return** $(b_2 \wedge b_3)$

Game $\mathcal{G}_3$: The game $\mathcal{G}_3$ differs from $\mathcal{G}_2$ when it aborts in Line 12 instead of returning the winning condition. This event only occurs when the adversary outputs a valid signature for the message m_{AH} alongside a secret s that is not a witness to the embedded statement Y' in the puzzle τ. We will show that this can only happen if the adversary breaks extractability of Π_{AS}. If this event does not occur, $\mathcal{G}_3$ does not differ from $\mathcal{G}_2$. The formal description of $\mathcal{G}_3$ is given in Figure 8.7.

Claim 6. *Let $\mathsf{Break}_{\mathsf{Extract}}$ be the event in Line 12, where $\mathcal{G}_3$ aborts. Then the adversary $\mathcal{A}$ was able to output a valid signature $\hat{\sigma}$ on the message m_{AH}, that does not allow extracting a valid witness s for the statement Y'. If $\mathsf{Break}_{\mathsf{Extract}}$ happens, we can build a reduction that breaks extractability of the adaptor signature scheme Π_{AS}. Therefore, the probability of $\mathsf{Break}_{\mathsf{Extract}}$ occurring is negligible, and it holds that* $\Pr[\mathcal{G}_3] \approx \Pr[\mathcal{G}_2]$.

Proof of Claim 6. During the execution of the $\mathsf{PSolver}$ protocol, the randomizable puzzle Z' and therefore the statement Y' are being rerandomized to a fresh looking version $(Z'', r') \leftarrow \Pi_{\mathsf{RP}}.\mathsf{PRand}(\mathsf{pp}, Z')$. The randomized statement Y'' is then embedded in the pre-signature $\tilde{\sigma}_{AH}$ and handed over to the adversary alongside the puzzle Z''. Evidently, any valid witness y'' to Y'' can be converted to a valid witness y' of Y' by knowing the randomization factor r'. Hence, $\mathsf{Break}_{\mathsf{Extract}}$ occurring must be due to an invalid witness y''. Based on this fact, we now give a reduction that uses $\mathcal{A}$

from game G_3 to break extractability of Π_{AS}. The reduction is given as input, a verification key vk^A, and has access to a pSign oracle that, on input, a message m and statement Y, returns a pre-signature $\tilde{\sigma}$ on m under the verification key vk^A as in Definition 14. The pSign oracle is efficient, as it follows the PPT pre-signing algorithm $\Pi_{\mathsf{AS}}.\mathsf{pSign}$. The reduction forwards the pSign oracle to $\mathcal{A}$ and runs $\mathcal{A}$ from $\mathcal{G}_3$ on input vk^A. The adversary $\mathcal{A}$ now first runs the $\mathsf{PPromise}$ protocol honestly, outputting a valid puzzle τ in Line 3 of $\mathcal{G}_3$. During the $\mathsf{PSolver}$ execution in $\mathcal{G}_3$ the reduction then parses the embedded statement from the puzzle **parse** $(Y'', c'') := Z''$, and queries the pSign oracle on the message m_{AH} and statement Y'' to receive a valid pre-signature $\tilde{\sigma}_{AH}$ under the verification key vk^A. The reduction then sends the pre-signature $\tilde{\sigma}_{AH}$ alongside the puzzle Z'' to $\mathcal{A}$ and completes the $\mathsf{PSolver}$ protocol with $\mathcal{A}$. Since we assume in Claim 6 that the event $\mathsf{Break}_{\mathsf{Extract}}$ happens with non-negligible probability, $\mathcal{A}$ eventually outputs a valid full signature σ^* in the $\mathsf{PSolver}$ protocol on the message m_{AH} under the verification key vk^A, that extracts to a witness $y'' = y' + r'$ such that $g^{y'} \neq Y'$. The reduction then uses σ^* to break the extractability property of the adaptor signature scheme Π_{AS}. More precisely, σ^* is a valid signature, no other oracle than the pSign oracle has been queried and the pre-signature and full signature do not extract to a valid statement in the relation $(Y', \mathsf{Extract}(\mathsf{vk}^A, \tilde{\sigma}_{AH}, \sigma^*, Y')) \notin \mathsf{Rel}$. Furthermore, the pSign oracle and $\mathcal{A}$ run in polynomial time, and by assumption of Claim 6, the event $\mathsf{Break}_{\mathsf{Extract}}$ happens with non-negligible probability. This reduction therefore breaks extractability of Π_{AS} also with non-negligible probability. This contradicts our assumption on the adaptor signature scheme Π_{AS} in Lemma 2. Hence, $\mathsf{Break}_{\mathsf{Extract}}$ can only occur with negligible probability, and it holds that $\Pr[\mathcal{G}_3] \approx \Pr[\mathcal{G}_2]$. □

Now that we have established that any verifying signature $\hat{\sigma}$ on the message m_{AH} needs to reveal a valid secret to the statement Y' in the puzzle τ, we will show that there can not exist an adversary with probability larger than negligible against the final game $\mathcal{G}_3$. We conclude the proof of Lemma 2 by demonstrating that any extractable witness s from the $\mathsf{PSolver}$ protocol allows opening the puzzle τ and outputting a valid signature σ_{HB}. Thus, given the stated assumptions, no adversary against the final game $\mathcal{G}_3$ can exist with a probability larger than negligible.

Claim 7. *If there exists a* PPT *adversary that wins game* $\mathcal{G}_3$ *with probability larger than negligible, then we can build a reduction that breaks pre-signature adaptability of the underlying adaptor signature scheme* Π_{AS}*. Hence, no* PPT *adversary against the final game* $\mathcal{G}_3$ *with probability larger than negligible exists.*

Proof of Claim 7. We prove Claim 7 with a reduction that uses $\mathcal{A}$ from game $\mathcal{G}_3$ to break pre-signature adaptability of Π_{AS}. The reduction generates a key pair $(\mathsf{vk}^A, \mathsf{sk}^A)$, and provides vk^A and a pSign oracle to $\mathcal{A}$ that on input a message m and statement Y returns a pre-signature $\tilde{\sigma}$ on m under the verification key vk^A by running the PPT algorithm $\Pi_{\mathsf{AS}}.\mathsf{pSign}(\mathsf{sk}^A, m, Y)$. First, $\mathcal{A}$ now generates a key pair for the adaptor signature scheme $(\mathsf{vk}^H, \mathsf{sk}^H)$ and LOE scheme $(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}})$ and two arbitrary messages m_{HB} and m_{AH}. The reduction then runs $\mathcal{A}$ on input vk^A to receive τ as a valid output of the $\mathsf{PPromise}$ protocol. The output τ contains a valid pre-signature $\tilde{\sigma}_{HB}$ on the message

m_{HB} under the verification key vk^H with an embedded statement $Y \in \mathsf{Rel}$, such that $\mathsf{pVrf}(\mathsf{vk}^H, m_{HB}, Y, \tilde{\sigma}_{HB}) = 1$. Additionally, τ contains the randomization factor r and a puzzle Z'' as in any honest execution of $\mathsf{PPromise}$.

In the $\mathsf{PSolver}$ protocol, the reduction then queries the pSign oracle on the statement Y'' contained in the puzzle Z'' and the message m_{AH} to receive a valid pre-signature $\tilde{\sigma}_{AH}$ under the verification key vk^A. The reduction sends the pre-signature $\tilde{\sigma}_{AH}$ and the puzzle Z'' to $\mathcal{A}$. The adversary $\mathcal{A}$ finally outputs a valid full signature on the message m_{AH} as its output of the $\mathsf{PSolver}$ protocol. The reduction then runs the $\mathsf{Extract}(\mathsf{vk}^A, \tilde{\sigma}_{AH}, \sigma_{AH})$ algorithm to receive a secret s and returns (σ^*, s) as its output of $\mathsf{PSolver}$. Since we assume that $\mathcal{A}$ wins the game $\mathcal{G}_3$ with non-negligible probability in Claim 7, σ^* is a valid signature under the verification key vk^A (b_2), and the Open algorithm does not return a valid full signature for the message m_{HB} on input the pre-signature $\tilde{\sigma}_{HB}$ and secret s (b_3).

From Line 12 in $\mathcal{G}_3$ it follows that any verifying signature σ^*, output by the $\mathsf{PSolver}$ protocol, also outputs a valid secret s to the statement Y' in τ, such that $Y' = g^s$. In the Open algorithm, the witness s is derandomized using the randomization factor r contained in the puzzle τ to receive a witness $y = y' - r$ for the statement. From the randomizable puzzle scheme (Section 2.5), it further follows that $Y' = Y \cdot g^r$. Therefore, given the randomization factor r, any valid witness for the embedded statement Y' can be derandomized to a valid witness for the original statement Y, and it holds that $Y = g^y$. It follows that y is a valid witness for the statement Y embedded in the pre-signature $\tilde{\sigma}_{HB}$ and the failing validation in b_3 needs to be due to the Adapt algorithm not returning a valid signature on the message m_{HB}.

This reduction breaks pre-signature adaptability of Π_{AS} with non-negligible probability since $\tilde{\sigma}_{HB}$ is a valid pre-signature on the message m_{HB}, that cannot be adapted to a valid full signature using a valid witness y for the embedded statement Y. The reduction is efficient since it only uses PPT algorithms from the underlying adaptor signature scheme Π_{AS} and, by assumption of Claim 7, $\mathcal{A}$ runs in polynomial time. We assumed by contradiction that $\mathcal{A}$ wins game $\mathcal{G}_3$ with non-negligible probability and showed that from this assumption we can build a reduction that breaks pre-signature adaptability of the adaptor signature scheme Π_{AS}. This is a contradiction, and thus, no adversary with a probability larger than negligible can exist against the game $\mathcal{G}_3$. □

Using Claim 7, we can finally conclude the proof of Lemma 2. We have shown through a series of game hops with negligible transitions that, under the stated assumptions, the probability for any PPT adversary winning the unlockability experiment $\mathsf{ExpUnlock}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ is negligibly close to the probability of winning the final game $\mathcal{G}_3$, and it holds that

$$\Pr\left[\mathsf{ExpUnlock}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}} = 1\right] \leq \Pr[\mathcal{G}_3 = 1] + \mathsf{negl}(\lambda)_1 + \mathsf{negl}(\lambda)_2.$$

From the assumption that there exists a PPT adversary $\mathcal{A}$ that wins the original game $\mathsf{ExpUnlock}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ with non-negligible probability, it follows that $\mathcal{A}$ also wins $\mathcal{G}_3$ with non-negligible probability. This contradicts Claim 7, and thus, no such polynomial-bound adversary can exist. It holds that A2L+ satisfies unlockability. □

8.3 Unforgeability

Lemma 3 (Unforgeability)**.** *We assume the existence of group description parameter* $gp = (\mathbb{G}, g, q)$*. Let* Rel *be a canonical hard relation with* PPT *sampling algorithm* $\mathsf{GenR}(1^\lambda)$*. Let* Π_{RP} *be a secure randomizable puzzle scheme according to Construction 1 that leverages an* IND-CCA*-secure LOE scheme* Π_E*. Let* NIZK *be a sound randomizable non-interactive zero-knowledge proof system, and let* Π_{AS} *be a secure adaptor signature scheme that achieves pre-signature correctness, extractability, unique extractability, unlinkability, pre-verify soundness, and pre-signature adaptability. Assuming the hardness of the OMDL, the A2L+ protocol satisfies unforgeability (c.f. Definition 23) in the LOE model.*

Figure 8.8: The first unforgeability game $\mathcal{G}_0(\lambda)$.

$\mathcal{G}_0(\lambda)$

$\mathcal{L} := \emptyset, \mathcal{Q} := 0$

$(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}}) \leftarrow \mathsf{Setup}(1^\lambda)$

$(\mathsf{vk}_1^H, m_1, \sigma_1), ..., (\mathsf{vk}_q^H, m_q, \sigma_q) \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{PP}(\cdot), \mathcal{O}\mathsf{PS}(\cdot)}(\tilde{\mathsf{ek}})$

$b_0 := \exists i \in [q]\ s.t.\ (\mathsf{vk}_i^H, \cdot) \in \mathcal{L} \wedge (\mathsf{vk}_i^H, m_i) \notin \mathcal{L} \wedge \mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1$

$b_1 := \forall i \in [q],\ (\mathsf{vk}_i^H, m_i) \in \mathcal{L} \wedge \mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1$

$b_2 := \bigwedge_{i,j \in [q], i \neq j} (\mathsf{vk}_i^H, m_i, \sigma_i) \neq (\mathsf{vk}_j^H, m_j, \sigma_j)$

$b_3 := (\mathcal{Q} \leq q - 1)$

return $b_0 \vee (b_1 \wedge b_2 \wedge b_3)$

$\mathcal{O}\mathsf{PP}(m)$

$(\mathsf{vk}^H, \mathsf{sk}^H) \leftarrow \Pi_{\mathsf{AS}}.\mathsf{KGen}(1^\lambda)$

$\mathcal{L} := \mathcal{L} \cup \{(\mathsf{vk}^H, m)\}$

$\perp \leftarrow \mathsf{PPromise}\Big(H(\tilde{\mathsf{dk}}, \mathsf{sk}^H, m), \mathcal{A}(\mathsf{vk}^H)\Big)$

$\mathcal{O}\mathsf{PS}(\mathsf{vk}^A, m')$

$(\sigma^*, s) \leftarrow \mathsf{PSolver}\Big(\mathcal{A}, H(\tilde{\mathsf{dk}}, \mathsf{vk}^A, m')\Big)$

if $\sigma^* \neq \perp$ **then** $\mathcal{Q} := \mathcal{Q} + 1$

Proof of Lemma 3. We prove Lemma 3 by showing through a series of game hops with negligible transitions that $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ can be reduced to the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}$ game under the stated assumptions. Glaeser et al. [GMM+22] showed by contradiction that any adversary winning the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}$ game with probability larger than

negligible can also break the OMDL problem, which is believed to be computationally hard in the LOE model. Hence, the probability of winning the $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ must be negligible in the LOE model, and the scheme satisfies the unforgeability notion under the stated assumptions.

Game $\mathcal{G}_0$: The first game $\mathcal{G}_0$ is the original $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ game in Figure 4.3. $\mathcal{G}_0$ simulates $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ perfectly, and it therefore holds that $\Pr\left[\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}} = 1\right] = \Pr[\mathcal{G}_0 = 1]$. The formal description of $\mathcal{G}_0$ is given in Figure 8.8.

Figure 8.9: The second unforgeability game $\mathcal{G}_1(\lambda)$.

$\mathcal{G}_1(\lambda)$

1 : $\mathcal{L} := \emptyset, \mathcal{Q} := 0$
2 : $(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}}) \leftarrow \mathsf{Setup}(1^\lambda)$
3 : $(\mathsf{vk}_1^H, m_1, \sigma_1), ..., (\mathsf{vk}_q^H, m_q, \sigma_q) \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{PP}(\cdot),\mathcal{O}\mathsf{PS}(\cdot)}(\tilde{\mathsf{ek}})$
4 : **if** $\exists i \in [q]$ *s.t.* $(\mathsf{vk}_i^H, \cdot) \in \mathcal{L} \wedge (\mathsf{vk}_i^H, m_i) \notin \mathcal{L} \wedge \mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1$ **then abort**
5 : $b_1 := \forall i \in [q],\ (\mathsf{vk}_i^H, m_i) \in \mathcal{L} \wedge \mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1$
6 : $b_2 := \bigwedge_{i,j \in [q], i \neq j} (\mathsf{vk}_i^H, m_i, \sigma_i) \neq (\mathsf{vk}_j^H, m_j, \sigma_j)$
7 : $b_3 := (\mathcal{Q} \leq q - 1)$
8 : **return** $(b_1 \wedge b_2 \wedge b_3)$

$\mathcal{O}\mathsf{PP}(m)$

1 : $(\mathsf{vk}^H, \mathsf{sk}^H) \leftarrow \Pi_{\mathsf{AS}}.\mathsf{KGen}(1^\lambda)$
2 : $\mathcal{L} := \mathcal{L} \cup \{(\mathsf{vk}^H, m)\}$
3 : $\perp \leftarrow \mathsf{PPromise}\left\langle H(\tilde{\mathsf{dk}}, \mathsf{sk}^H, m), \mathcal{A}(\mathsf{vk}^H)\right\rangle$

$\mathcal{O}\mathsf{PS}(\mathsf{vk}^A, m')$

1 : $(\sigma^*, s) \leftarrow \mathsf{PSolver}\left\langle \mathcal{A}, H(\tilde{\mathsf{dk}}, \mathsf{vk}^A, m')\right\rangle$
2 : **if** $\sigma^* \neq \perp$ **then** $\mathcal{Q} := \mathcal{Q} + 1$

Game $\mathcal{G}_1$: The second game $\mathcal{G}_1$ differs from game $\mathcal{G}_0$ when it aborts in Line 4 instead of returning the winning condition. This event occurs only when the adversary outputs a valid signature on a message that has not been queried by any oracle, thereby producing a valid forgery. This game hop does not differ from $\mathcal{G}_0$ if this event does not occur. The formal description of $\mathcal{G}_1$ is given in Figure 8.9.

Claim 8. *Let* $\mathsf{Break}_{\mathit{Extract}}$ *be the event in Line 4, where* $\mathcal{G}1$ *aborts. Then, the adversary* $\mathcal{A}$ *found a valid signature forgery* σ_i *on a message* m_i *that has not been queried by any oracle. If* $\mathsf{Break}_{\mathit{Extract}}$ *happens, then we can build a reduction that can break extractability of the underlying adaptor signature scheme* Π_{AS}*. Therefore,* $\mathsf{Break}_{\mathit{Extract}}$ *can occur only with negligible probability. Since* $\Pr[\mathcal{G}_1 \wedge \neg \mathsf{Break}_{\mathit{Extract}}] = \Pr[\mathcal{G}_0 \wedge \neg \mathsf{Break}_{\mathit{Extract}}]$ *and the event* $\mathsf{Break}_{\mathit{Extract}}$ *occurring is negligible,* $\mathcal{G}_0$ *and* $\mathcal{G}_1$ *need to be indistinguishable. It holds that* $\Pr[\mathcal{G}_1] \approx \Pr[\mathcal{G}_0]$.

Proof of Claim 8. We prove Claim 8 with a reduction that uses $\mathcal{A}$ from game $\mathcal{G}_1$ to break extractability of the underlying adaptor signature scheme Π_{AS}. The reduction receives as input a number of challenge verification keys vk_i and a pSign oracle that, on input, a message m and statement Y, returns a pre-signature $\widetilde{\sigma}$ on m under the verification key vk_i as in Definition 14. The pSign oracle is efficient, as it follows the PPT pre-signing algorithm $\Pi_{\mathsf{AS}}.\mathsf{pSign}$. The reduction forwards the pSign oracle to $\mathcal{A}$ by answering a polynomially bounded number of queries. Furthermore, it simulates $\mathcal{G}_1$, with the respective oracles $\mathcal{O}\mathsf{PP}$ and $\mathcal{O}\mathsf{PS}$ perfectly from the view of $\mathcal{A}$. The adversary $\mathcal{A}$ now queries the $\mathcal{O}\mathsf{PP}$ oracle in $\mathcal{G}_1$ up to q times on messages of its choice. During each execution of the $\mathsf{PPromise}$ protocol within the $\mathcal{O}\mathsf{PP}$ oracle, instead of computing the pre-signatures itself, the reduction queries the pSign oracle to obtain valid pre-signatures on the respective messages under the verification keys vk_i. Additionally, $\mathcal{A}$ queries the $\mathcal{O}\mathsf{PS}$ oracle on $q-1$ messages of its choice to receive valid signatures under its own secret key sk^A. Since we assume in *Claim* 8 that the event $\mathsf{Break}_{\mathsf{Extract}}$ occurs with non-negligible probability, $\mathcal{A}$ eventually outputs a valid signature σ_i for a message m_i under one of the verification keys vk_i for which the $\mathcal{O}\mathsf{PP}$ and therefore also the pSign oracle has not been queried. Finally, the reduction can use σ_i to break extractability of the adaptor signature scheme Π_{AS}, since the signature σ_i is valid under the verification key vk_i and neither was a Sign oracle provided nor has the pSign oracle been queried on the message m_i. In Claim 8, we assumed that $\mathcal{A}$ is a PPT adversary and that $\mathsf{Break}_{\mathsf{Extract}}$ happens with non-negligible probability. Hence, the presented reduction is efficient and breaks extractability with non-negligible probability. This contradicts our assumption about the adaptor signature scheme Π_{AS} in Lemma 3. Thus, the event $\mathsf{Break}_{\mathsf{Extract}}$ can only occur with negligible probability, and it holds that $\Pr[\mathcal{G}_1] \approx \Pr[\mathcal{G}_0]$. □

Game $\mathcal{G}_2$: The game $\mathcal{G}_2$ differs from $\mathcal{G}_1$ in Lines 5 and 7, where instead of requiring that all $(\mathsf{vk}_i, m_i, \sigma_1)$ tuples are different, we require that all q outputs of the $\mathsf{Extract}$ algorithm are different. Effectively, this means all witnesses are different. The formal description of $\mathcal{G}_2$ is given in Figure 8.10.

Claim 9. *Assuming the puzzle promise protocol samples the witnesses randomly from a uniform distribution. If the* $\mathsf{Extract}$ *algorithm of any two executions* $\mathsf{Extract}(\mathsf{vk}_i, \widetilde{\sigma}_i, \sigma_i, Y_i)$ *outputs the same witness, while the message signature tuples* $(\mathsf{vk}_i, m_i, \sigma_i)$ *are different, then we can build a reduction that breaks unique extractability of the underlying adaptor signature scheme* Π_{AS} *with probability larger than negligible. Hence, the two games must be indistinguishable* $\Pr[\mathcal{G}_2] \approx \Pr[\mathcal{G}_1]$.

Figure 8.10: The third unforgeability game $\mathcal{G}_2(\lambda)$.

$\mathcal{G}_2(\lambda)$

$\mathcal{L} := \emptyset, \mathcal{S} := \emptyset, \mathcal{Q} := 0$
$(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}}) \leftarrow \mathsf{Setup}(1^\lambda)$
$(\mathsf{vk}_1^H, m_1, \sigma_1, \widetilde{\sigma}_1, Y_1), ..., (\mathsf{vk}_q^H, m_q, \sigma_q, \widetilde{\sigma}_q, Y_q) \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{PP}(\cdot), \mathcal{O}\mathsf{PS}(\cdot)}(\tilde{\mathsf{ek}})$
if $\exists i \in [q]$ *s.t.* $(\mathsf{vk}_i^H, \cdot) \in \mathcal{L} \wedge (\mathsf{vk}_i^H, m_i) \notin \mathcal{L} \wedge \mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1$ **then abort**
$\forall i \in [q], \mathcal{S} := \mathcal{S} \cup \{\mathsf{Extract}(\mathsf{vk}_i, \widetilde{\sigma}_i, \sigma_i, Y_i)\}$
$b_1 := \forall i \in [q],\ (\mathsf{vk}_i^H, m_i) \in \mathcal{L} \wedge \mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1$
$b_2 := \forall y_i, y_j \in \mathcal{S}, i \neq j : y_i \neq y_j$
$b_3 := (\mathcal{Q} \leq q - 1)$
return $(b_1 \wedge b_2 \wedge b_3)$

$\mathcal{O}\mathsf{PP}(m)$

$(\mathsf{vk}^H, \mathsf{sk}^H) \leftarrow \Pi_{\mathsf{AS}}.\mathsf{KGen}(1^\lambda)$
$\mathcal{L} := \mathcal{L} \cup \{(\mathsf{vk}^H, m)\}$
$\perp \leftarrow \mathsf{PPromise}\Big\langle H(\tilde{\mathsf{dk}}, \mathsf{sk}^H, m), \mathcal{A}(\mathsf{vk}^H)\Big\rangle$

$\mathcal{O}\mathsf{PS}(\mathsf{vk}^A, m')$

$(\sigma^*, s) \leftarrow \mathsf{PSolver}\Big\langle \mathcal{A}, H(\tilde{\mathsf{dk}}, \mathsf{vk}^A, m')\Big\rangle$
if $\sigma^* \neq \perp$ **then** $\mathcal{Q} := \mathcal{Q} + 1$

Proof of Claim 9. We prove Claim 9 with a reduction that uses $\mathcal{A}$ from game $\mathcal{G}_2$ to create an adaptor signature scheme with malleable pre-signatures, similar to Figure 5.1 by Gerhart et al. [GSST24], to break unique extractability of the underlying adaptor signature scheme Π_{AS}. The reduction receives as inputs the verification keys vk_i and a pSign oracle that, on input, a message m and statement Y, returns a pre-signature $\widetilde{\sigma}$ on m under the verification key vk_i as in Definition 14. The pSign oracle is efficient, as it follows the PPT pre-signing algorithm $\Pi_{\mathsf{AS}}.\mathsf{pSign}$. During each $\mathsf{PPromise}$ protocol execution in the $\mathcal{O}\mathsf{PP}$ oracle, the reduction queries the pSign oracle with the verification key vk_i instead of computing the pre-signature itself. The reduction initially creates the key pair $(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}})$, and runs $\mathcal{A}$ on input $\tilde{\mathsf{ek}}$. The adversary $\mathcal{A}$ now queries the $\mathcal{O}\mathsf{PP}$ oracle q times to receive q valid pre-signatures $\widetilde{\sigma}_q$ from the pSign oracle with embedded statements $Y_q \in \mathsf{Rel}$ on the messages m_q of its choice. Furthermore, $\mathcal{A}$ queries the $\mathcal{O}\mathsf{PP}$ oracle $q-1$ times to receive valid secrets s_{q-1} for the embedded challenges in $\widetilde{\sigma}_q$. On the i-th query $\mathcal{A}$ then runs the Adapt algorithm of Π_{AS} on input the pre-signature $\widetilde{\sigma}_i$ and secret s_i, and outputs a valid signature σ_{i1} on the message m_i using the pre-signature $\widetilde{\sigma}_i$.

We assume by contradiction in Claim 9 that $\mathcal{A}$ outputs two distinct valid signatures $(\sigma_{i1}, \sigma_{i2})$ that extract to the same witness s_i with non-negligible probability. Since in the PPromise protocol, the embedded secrets s_i get sampled uniformly at random from a large distribution, the probability that any two statements Y_i in two different pre-signatures $\tilde{\sigma}_i$ are the same is negligible. Therefore, when two signatures extract to the same witness, they both need to be an adapted signature using the same pre-signature. Hence, in the reduction, $\mathcal{A}$ runs the $\Pi_{\mathsf{AS}}.\mathsf{Adapt}$ algorithm again on the i-th pre-signature $\tilde{\sigma}_i$ to receive another signature σ_{i2}, similar to the adaptor signature scheme with malleable pre-signatures in Figure 5.1. The reduction finally outputs the pre-signature $\tilde{\sigma}_i$, message m_i, and the two signatures σ_{i1} and σ_{i2}, to break the unique extractability property of Π_{AS}. More precisely, running the $\Pi_{\mathsf{AS}}.\mathsf{Extract}$ algorithm on both signatures σ_{i1} and σ_{i2} will result in the same witness, and the embedded statement is in the relation $Y_i \in \mathsf{Rel}$, since it has been honestly sampled in the PPromise protocol. The reduction is efficient as $\mathcal{A}$ runs in polynomial time and by assumption of Claim 9 has a non-negligible success probability. Hence, this reduction breaks unique extractability of Π_{AS} also with non-negligible probability, which contradicts our assumption on the adaptor signature scheme Π_{AS} in Lemma 3. Evidently, any pre-signature $\tilde{\sigma}_i$ with an embedded witness Y_i must be a commitment to a single full signature σ_i, and the probability that two signatures differ while they extract to the same witness is the same as the probability of two randomly sampled witnesses being identical. In the PPromise protocol in $\mathcal{G}_2$, the witnesses get sampled uniformly at random from a large distribution. Hence, the probability of two witnesses being identical is negligible. It holds that $\mathcal{G}_1$ and $\mathcal{G}_2$ are indistinguishable. □

Game $\mathcal{G}_3$: $\mathcal{G}_3$ differs from game $\mathcal{G}_2$ when it aborts in Line 5. This event only occurs when the adversary outputs a valid signature that does not extract to a valid witness for the statement Y_i. This game hop does not differ from $\mathcal{G}_2$ if this event does not occur. The formal description of $\mathcal{G}_3$ is given in Figure 8.11.

Claim 10. *Let $\mathsf{Break}_{\mathsf{Extract}}$ be the event in Line 5, where $\mathcal{G}_3$ aborts. Then, the adversary $\mathcal{A}$ found a valid signature that does not allow extracting a valid witness for the embedded statement Y_i. If $\mathsf{Break}_{\mathsf{Extract}}$ happens, then we can build a reduction that can break extractability of the underlying adaptor signature scheme Π_{AS}. Therefore, $\mathsf{Break}_{\mathsf{Extract}}$ can occur only with negligible probability. Since $\Pr[\mathcal{G}_3 \wedge \neg\mathsf{Break}_{\mathsf{Extract}}] = \Pr[\mathcal{G}_2 \wedge \neg\mathsf{Break}_{\mathsf{Extract}}]$ and the event $\mathsf{Break}_{\mathsf{Extract}}$ occurring is negligible, $\mathcal{G}_2$ and $\mathcal{G}_3$ need to be indistinguishable. It holds that $\Pr[\mathcal{G}_3] \approx \Pr[\mathcal{G}_2]$.*

Proof of Claim 10. We prove Claim 10 with a reduction that uses $\mathcal{A}$ from game $\mathcal{G}_3$ to break extractability of the underlying adaptor signature scheme Π_{AS}. The reduction receives as inputs the verification keys vk_i and a pSign oracle that, on input, a message m and statement Y, returns a pre-signature $\tilde{\sigma}$ on m under the verification key vk_i as in Definition 14. The pSign oracle is efficient, as it follows the PPT pre-signing algorithm $\Pi_{\mathsf{AS}}.\mathsf{pSign}$. The reduction initially creates the key pair $(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}})$, and runs $\mathcal{A}$ on input $\tilde{\mathsf{ek}}$. The adversary now interacts with the $\mathcal{O}\mathsf{PP}$ oracle as in $\mathcal{G}_3$ to receive

Figure 8.11: The fourth unforgeability game $\mathcal{G}_3(\lambda)$.

$\mathcal{G}_3(\lambda)$

1 : $\mathcal{L} := \emptyset, \mathcal{S} := \emptyset, \mathcal{Q} := 0$
2 : $(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}}) \leftarrow \mathsf{Setup}(1^\lambda)$
3 : $(\mathsf{vk}_1^H, m_1, \sigma_1, \widetilde{\sigma}_1, Y_1), ..., (\mathsf{vk}_q^H, m_q, \sigma_q, \widetilde{\sigma}_q, Y_q) \leftarrow \mathcal{A}^{\mathcal{O}\mathsf{PP}(\cdot), \mathcal{O}\mathsf{PS}(\cdot)}(\tilde{\mathsf{ek}})$
4 : **if** $\exists i \in [q]$ *s.t.* $(\mathsf{vk}_i^H, \cdot) \in \mathcal{L} \wedge (\mathsf{vk}_i^H, m_i) \notin \mathcal{L} \wedge \mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1$ **then abort**
5 : **if** $\exists i \in [q]$ *s.t.* $\mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1 \wedge g^{\mathsf{Extract}(\mathsf{vk}_i, \widetilde{\sigma}_i, \sigma_i, Y_i)} \neq Y_i$ **then abort**
6 : $\forall i \in [q], \mathcal{S} := \mathcal{S} \cup \{\mathsf{Extract}(\mathsf{vk}_i, \widetilde{\sigma}_i, \sigma_i, Y_i)\}$
7 : $b_1 := \forall i \in [q],\ (\mathsf{vk}_i^H, m_i) \in \mathcal{L} \wedge \mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1$
8 : $b_2 := \forall y_i, y_j \in \mathcal{S}, i \neq j : y_i \neq y_j$
9 : $b_3 := (\mathcal{Q} \leq q - 1)$
10 : **return** $(b_1 \wedge b_2 \wedge b_3)$

$\mathcal{O}\mathsf{PP}(m)$

1 : $(\mathsf{vk}^H, \mathsf{sk}^H) \leftarrow \Pi_{\mathsf{AS}}.\mathsf{KGen}(1^\lambda)$
2 : $\mathcal{L} := \mathcal{L} \cup \{(\mathsf{vk}^H, m)\}$
3 : $\perp \leftarrow \mathsf{PPromise}\Big\langle H(\tilde{\mathsf{dk}}, \mathsf{sk}^H, m), \mathcal{A}(\mathsf{vk}^H) \Big\rangle$

$\mathcal{O}\mathsf{PS}(\mathsf{vk}^A, m')$

1 : $(\sigma^*, s) \leftarrow \mathsf{PSolver}\Big\langle \mathcal{A}, H(\tilde{\mathsf{dk}}, \mathsf{vk}^A, m') \Big\rangle$
2 : **if** $\sigma^* \neq \perp$ **then** $\mathcal{Q} := \mathcal{Q} + 1$

valid pre-signatures $\widetilde{\sigma}_i$ on messages m_i of its choice. The reduction computes the pre-signatures during the $\mathsf{PPromise}$ protocol by calling the pSign oracle instead of computing them itself. The adversary then queries the $\mathcal{O}\mathsf{PS}$ oracle to run the $\mathsf{PSolver}$ protocol and receive a valid witness s_i for the statement embedded in the pre-signature $\widetilde{\sigma}_i$. With the knowledge of s_i, $\mathcal{A}$ runs the $\Pi_{\mathsf{AS}}.\mathsf{Adapt}$ algorithm to receive valid full signatures σ_i on the messages m_i. Since we assume in Claim 10, that $\mathsf{Break}_{\mathsf{Extract}}$ happens with non-negligible probability, $\mathcal{A}$ eventually outputs a valid full signature σ_i on a message m_i under the verification key vk_i that does not allow extracting a valid witness to the embedded statement Y_i using the corresponding pre-signature $\widetilde{\sigma}_i$, such that $Y_i \neq g^{\mathsf{Extract}(\mathsf{vk}_i, \widetilde{\sigma}_i, \sigma_i, Y_i)}$. This contradicts the extractability property of the adaptor signature scheme Π_{AS}, since σ_i is a valid signature, no Sign oracle has been provided, and no valid witness can be extracted. We assumed by contradiction that $\mathcal{A}$ runs in polynomial time and $\mathsf{Break}_{\mathsf{Extract}}$ happens with non-negligible probability. Thus, the given reduction is an efficient algorithm that also breaks extractability of Π_{AS} with non-negligible probability, violating the assumption in Lemma 3. Hence,

the event $\mathsf{Break}_{\mathsf{Exctract}}$ can only occur with negligible probability, we conclude that $\Pr[\mathcal{G}_3] \approx \Pr[\mathcal{G}_2]$. □

We will now give a reduction from game $\mathcal{G}_3$ to the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}$ experiment (Figure 4.4), which, as Glaeser et al. [GMM+22] showed, can be further reduced to the OMDL problem, given that the encryption scheme Π_{E} is IND-CCA-secure.

Claim 11. *If there exists an efficient adversary that wins game $\mathcal{G}_3$ with probability larger than negligible, then we can build a reduction that breaks $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{E},q}$.*

Proof of Claim 11. The reduction is given $(c_1, h_1), \ldots, (c_i, h_i)$ as in $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}$. It first generates the keys $(\tilde{\mathsf{ek}}, \tilde{\mathsf{dk}}) \leftarrow \mathsf{Setup}(1^\lambda)$ and $(\mathsf{vk}_i^H, \mathsf{sk}_i^H)$ and starts $\mathcal{A}$ on input $\tilde{\mathsf{ek}}$. For each $\mathcal{O}\mathsf{PP}$ query, it uses a different challenge h_i, to generate the pre-signatures. Note that in $\mathcal{G}_3$, by definition, not all embedded witnesses are different. We do however assume that in the $\mathsf{PPromise}$ algorithm the statement witness pairs $(Y, y) \leftarrow \mathsf{Rel.GenR}(1^\lambda)$ are sampled uniformly at random from a large distribution and thus are unique with overwhelming probability. When $\mathcal{A}$ now queries the $\mathcal{O}\mathsf{PS}$ oracle, the reduction returns the adapted full signature as the output of $\mathcal{O}^{\mathsf{A^2L}}_{\mathsf{dk},\Pi_{\mathsf{E}},\Pi_{\mathsf{AS}}}$ run on input $(\mathsf{vk}^A, m_i, h_i, c_i, \widetilde{\sigma}_i)$. The $\mathcal{O}\mathsf{PS}$ and $\mathcal{O}^{\mathsf{A^2L}}_{\mathsf{dk},\Pi_{\mathsf{E}},\Pi_{\mathsf{AS}}}$ oracles return $\perp$ in exactly the same cases, consequently both $\mathcal{A}$ and the reduction return at most $q-1$ non-$\perp$ oracle queries. Since we assume by contradiction that $\mathcal{A}$ wins $\mathcal{G}_3$ with non-negligible probability, $\mathcal{A}$ outputs q valid tuples $\mathsf{Vf}(\mathsf{vk}_i^H, m_i, \sigma_i) = 1$ according to the winning condition b_1. The reduction now computes $r_i \leftarrow \Pi_{\mathsf{AS}}.\mathsf{Extract}(\mathsf{vk}_i^H, \widetilde{\sigma}_i, \sigma_i, Y_i)\ \forall i \in [q]$ and outputs those values to win the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}$ experiment. By the winning condition b_2 of $\mathcal{G}_3$, all r_i must be different when $\mathcal{A}$ completes successfully. Hence, it holds that $g^{r_i} = h_i\ \forall i \in [q]$. This breaks OM-CCA-A2L-security of the underlying IND-CCA-secure encryption scheme Π_{E}, and thus, no such efficient adversary can exist with non-negligible probability. □

We are now ready to conclude our proof of Lemma 3. Through a series of game hops, we showed that the probability of winning the unlockability game $\mathsf{ExpUnlock}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ is negligibly close to the probability of winning the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}$ experiment, and it holds that

$$\Pr\left[\mathsf{ExpUnlock}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}} = 1\right] \leq \Pr\left[\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q} = 1\right] + \mathsf{negl}(\lambda)_1 + \mathsf{negl}(\lambda)_2 + \mathsf{negl}(\lambda)_3.$$

We assumed by contradiction that an efficient adversary $\mathcal{A}$ winning the $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ game with non-negligible probability exists. This implies that $\mathcal{A}$ also wins the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}$ experiment with non-negligible probability. Glaeser et al. [GMM+22] showed that, assuming the encryption scheme is IND-CCA secure, the $\mathsf{OM\text{-}CCA\text{-}A2L}^{\mathcal{A}}_{\Pi_{\mathsf{E}},q}$ experiment can be reduced to the OMDL problem in the LOE model. Therefore, the existence of an efficient adversary $\mathcal{A}$ that breaks the OM-CCA-A2L-security of the underlying encryption scheme Π_{E} contradicts the hardness assumption of the OMDL. Hence, under the stated assumptions, no efficient adversary against $\mathsf{ExpUnforg}^{\mathcal{A}}_{\Pi_{\mathsf{BCS}}}$ exists, and A2L+ satisfies unforgeability. □

We can now conclude our proof of Theorem 1. In the preceding sections, we have proven that the A2L+ protocol satisfies the three security properties: unforgeability, selective-failure blindness, and unlockability. Each property was proven using game-based techniques established in the respective lemmas. Together Lemma 1, Lemma 2, and Lemma 3 provide a complete security analysis of the A2L+ scheme, thereby proving the overall security as stated in Theorem 1. Thus, A2L+ is a secure blind conditional signature scheme under the stated assumptions.

9 Conclusion

This thesis investigated the security of blind conditional signatures, the cryptographic core of coin-mixing protocols. We identified and addressed a previously known security flaw that rendered the A2L+ scheme insecure with respect to the unforgeability security notion. Additionally, we reviewed the current security definitions and introduced selective-failure blindness, which ensures blindness in case of aborts. Following this, we presented an improved construction of the A2L+ scheme that ensures blindness even in case the receiver and the Hub collude and demonstrated its security through game-based proofs.

The main findings of this thesis are that the identified security flaw can be mitigated by choosing an adaptor signature scheme that satisfies the stronger security definitions according to Gerhart et al. [GSST24]. Most notably an adaptor signature scheme additionally must satisfy unique extractability to fulfill the unforgeability security property. Our proposed definition of selective failure blindness and improved scheme contribute to the development of more robust coin-mixing protocols, ensuring higher security and privacy in blockchain transactions.

While our work provides significant improvements, future research needs to address some limitations. One limitation is the required coin-locking mechanism, for which Glaeser et al. [GMM+22] suggested a payment channel setup. Payment channels, on the other hand, are not available on all blockchains. Future research could therefore explore various coin-locking mechanisms.

Furthermore, blind conditional signatures could potentially lay the groundwork for a cross-chain coin-mixing protocol. All protocol phases do not need to be run on a single blockchain. The PPromise and PSolver algorithms could be executed on different chains. Hence, future work could construct a cross-chain protocol and analyze its security based on the BCS security properties presented in this work.

On a more foundational level, this thesis showed that the presented security properties are built upon a set of assumptions that do not hold in all circumstances. For instance, Glaeser et al. [GMM+22] assumed that there is no collusion between the Hub and another party. However, there might be cases in which we need a scheme that is secure in case this assumption does not hold. We showed that we can construct a scheme that is secure against a collusion between the receiver Bob and the Hub by adding the randomizable NIZK in our improved construction. The security properties do however not cover any collusion. Future research could therefore investigate more robust security frameworks that address challenges posed by colluding adversaries. This involves developing models that remain secure in a multi-adversary setting, thereby enhancing security in more complex adversarial environments.

In conclusion, this thesis has made substantial contributions to the field of coin-

mixing by addressing a critical vulnerability in the A2L+ protocol by Glaeser et al. [GMM+22]. Our findings highlight the importance of rigorous security definitions and robust constructions, paving the way for future advancements in the field.

Bibliography

[ADKL19] Ananth, Prabhanjan ; Deshpande, Apoorvaa ; Kalai, Yael T. ; Lysyanskaya, Anna: Fully Homomorphic NIZK and NIWI Proofs. In: Hofheinz, Dennis (Hrsg.) ; Rosen, Alon (Hrsg.): *TCC 2019: 17th Theory of Cryptography Conference, Part II* Bd. 11892. Nuremberg, Germany : Springer, Cham, Switzerland, Dezember 1–5, 2019 (Lecture Notes in Computer Science), S. 356–385

[AEE+21] Aumayr, Lukas ; Ersoy, Oguzhan ; Erwig, Andreas ; Faust, Sebastian ; Hostáková, Kristina ; Maffei, Matteo ; Moreno-Sanchez, Pedro ; Riahi, Siavash: Generalized Channels from Limited Blockchain Scripts and Adaptor Signatures. In: Tibouchi, Mehdi (Hrsg.) ; Wang, Huaxiong (Hrsg.): *Advances in Cryptology – ASIACRYPT 2021, Part II* Bd. 13091. Singapore : Springer, Cham, Switzerland, Dezember 6–10, 2021 (Lecture Notes in Computer Science), S. 635–664

[BCC+09] Belenkiy, Mira ; Camenisch, Jan ; Chase, Melissa ; Kohlweiss, Markulf ; Lysyanskaya, Anna ; Shacham, Hovav: Randomizable Proofs and Delegatable Anonymous Credentials. In: Halevi, Shai (Hrsg.): *Advances in Cryptology – CRYPTO 2009* Bd. 5677. Santa Barbara, CA, USA : Springer, Berlin, Heidelberg, Germany, August 16–20, 2009 (Lecture Notes in Computer Science), S. 108–125

[BFP21] Bauer, Balthazar ; Fuchsbauer, Georg ; Plouviez, Antoine: The One-More Discrete Logarithm Assumption in the Generic Group Model. In: Tibouchi, Mehdi (Hrsg.) ; Wang, Huaxiong (Hrsg.): *Advances in Cryptology – ASIACRYPT 2021, Part IV* Bd. 13093. Singapore : Springer, Cham, Switzerland, Dezember 6–10, 2021 (Lecture Notes in Computer Science), S. 587–617

[BNPS03] Bellare, Mihir ; Namprempre, Chanathip ; Pointcheval, David ; Semanko, Michael: The One-More-RSA-Inversion Problems and the Security of Chaum's Blind Signature Scheme. In: *Journal of Cryptology* 16 (2003), Juni, Nr. 3, S. 185–215. `http://dx.doi.org/10.1007/s00145-002-0120-1`. – DOI 10.1007/s00145–002–0120–1

[CL15] Castagnos, Guilhem ; Laguillaumie, Fabien: Linearly Homomorphic Encryption from DDH. In: Nyberg, Kaisa (Hrsg.): *Topics in Cryptology – CT-RSA 2015* Bd. 9048. San Francisco, CA, USA : Springer, Cham,

Switzerland, April 20–24, 2015 (Lecture Notes in Computer Science), S. 487–505

[CNs07] Camenisch, Jan ; Neven, Gregory ; shelat, abhi: Simulatable Adaptive Oblivious Transfer. In: Naor, Moni (Hrsg.): *Advances in Cryptology – EUROCRYPT 2007* Bd. 4515. Barcelona, Spain : Springer, Berlin, Heidelberg, Germany, Mai 20–24, 2007 (Lecture Notes in Computer Science), S. 573–590

[DMP88] De Santis, Alfredo ; Micali, Silvio ; Persiano, Giuseppe: Non-Interactive Zero-Knowledge Proof Systems. In: Pomerance, Carl (Hrsg.): *Advances in Cryptology – CRYPTO'87* Bd. 293. Santa Barbara, CA, USA : Springer, Berlin, Heidelberg, Germany, August 16–20, 1988 (Lecture Notes in Computer Science), S. 52–72

[DOY22] Dai, Wei ; Okamoto, Tatsuaki ; Yamamoto, Go: *Stronger Security and Generic Constructions for Adaptor Signatures*. Cryptology ePrint Archive, Report 2022/1687, 2022. – https://eprint.iacr.org/2022/1687

[DS21] Deuber, Dominic ; Schröder, Dominique: CoinJoin in the Wild - An Empirical Analysis in Dash. In: Bertino, Elisa (Hrsg.) ; Shulman, Haya (Hrsg.) ; Waidner, Michael (Hrsg.): *ESORICS 2021: 26th European Symposium on Research in Computer Security, Part II* Bd. 12973. Darmstadt, Germany : Springer, Cham, Switzerland, Oktober 4–8, 2021 (Lecture Notes in Computer Science), S. 461–480

[FS09] Fischlin, Marc ; Schröder, Dominique: Security of Blind Signatures under Aborts. In: Jarecki, Stanislaw (Hrsg.) ; Tsudik, Gene (Hrsg.): *PKC 2009: 12th International Conference on Theory and Practice of Public Key Cryptography* Bd. 5443. Irvine, CA, USA : Springer, Berlin, Heidelberg, Germany, März 18–20, 2009 (Lecture Notes in Computer Science), S. 297–316

[GM84] Goldwasser, Shafi ; Micali, Silvio: Probabilistic Encryption. In: *Journal of Computer and System Sciences* 28 (1984), Nr. 2, S. 270–299

[GMM+22] Glaeser, Noemi ; Maffei, Matteo ; Malavolta, Giulio ; Moreno-Sanchez, Pedro ; Tairi, Erkan ; Thyagarajan, Sri Aravinda K.: Foundations of Coin Mixing Services. In: Yin, Heng (Hrsg.) ; Stavrou, Angelos (Hrsg.) ; Cremers, Cas (Hrsg.) ; Shi, Elaine (Hrsg.): *ACM CCS 2022: 29th Conference on Computer and Communications Security*. Los Angeles, CA, USA : ACM Press, November 7–11, 2022, S. 1259–1273

[GMR88] Goldwasser, Shafi ; Micali, Silvio ; Rivest, Ronald L.: A Digital Signature Scheme Secure Against Adaptive Chosen-message Attacks. In: *SIAM Journal on Computing* 17 (1988), April, Nr. 2, S. 281–308

[Gro04] Groth, Jens: Rerandomizable and Replayable Adaptive Chosen Ciphertext Attack Secure Cryptosystems. In: [TCC 204], S. 152–170

[GS08] Groth, Jens ; Sahai, Amit: Efficient Non-interactive Proof Systems for Bilinear Groups. In: Smart, Nigel P. (Hrsg.): *Advances in Cryptology – EUROCRYPT 2008* Bd. 4965. Istanbul, Turkey : Springer, Berlin, Heidelberg, Germany, April 13–17, 2008 (Lecture Notes in Computer Science), S. 415–432

[GSST24] Gerhart, Paul ; Schröder, Dominique ; Soni, Pratik ; Thyagarajan, Sri A.: Foundations of Adaptor Signatures. In: Joye, Marc (Hrsg.) ; Leander, Gregor (Hrsg.): *Advances in Cryptology – EUROCRYPT 2024*. Cham : Springer Nature Switzerland, 2024. – ISBN 978–3–031–58723–8, S. 161–189

[HAB+17] Heilman, Ethan ; Alshenibr, Leen ; Baldimtsi, Foteini ; Scafuro, Alessandra ; Goldberg, Sharon: TumbleBit: An Untrusted Bitcoin-Compatible Anonymous Payment Hub. In: *ISOC Network and Distributed System Security Symposium – NDSS 2017*. San Diego, CA, USA : The Internet Society, Februar 26 – März 1, 2017

[KL14] Katz, Jonathan ; Lindell, Yehuda: *Introduction to Modern Cryptography*. Third. Chapman and Hall, CRC Press, 2014. – ISBN 978–0815354369

[Ped92] Pedersen, Torben P.: Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. In: Feigenbaum, Joan (Hrsg.): *Advances in Cryptology – CRYPTO'91* Bd. 576. Santa Barbara, CA, USA : Springer, Berlin, Heidelberg, Germany, August 11–15, 1992 (Lecture Notes in Computer Science), S. 129–140

[Poe17] Poelstra, Andrew: *Scriptless scripts*. `https://download.wpsoftware.net/bitcoin/wizardry/mw-slides/2017-03-mit-bitcoin-expo/slides.pdf`. Version: 2017. – Accessed: 2022-07-01

[PSS19] Pertsev, Alexey ; Semenov, Roman ; Storm, Roman: *Tornado Cash Privacy Solution Version 1.4*. `https://berkeley-defi.github.io/assets/material/Tornado%20Cash%20Whitepaper.pdf`, 2019. – Accessed: 2024-05-15

[RMK14] Ruffing, Tim ; Moreno-Sanchez, Pedro ; Kate, Aniket: CoinShuffle: Practical Decentralized Coin Mixing for Bitcoin. In: Kutylowski, Miroslaw (Hrsg.) ; Vaidya, Jaideep (Hrsg.): *ESORICS 2014: 19th European Symposium on Research in Computer Security, Part II* Bd. 8713. Wroclaw, Poland : Springer, Cham, Switzerland, September 7–11, 2014 (Lecture Notes in Computer Science), S. 345–364

[Sch91] Schnorr, Claus-Peter: Efficient Signature Generation by Smart Cards. In: *Journal of Cryptology* 4 (1991), Januar, Nr. 3, S. 161–174. `http://dx.doi.org/10.1007/BF00196725`. – DOI 10.1007/BF00196725

[TCC 204] Naor, Moni (Hrsg.): *TCC 2004: 1st Theory of Cryptography Conference*. Bd. *2951*. Cambridge, MA, USA : Springer, Berlin, Heidelberg, Germany, Februar 19–21, 2004 (Lecture Notes in Computer Science)

[TMM21] Tairi, Erkan ; Moreno-Sanchez, Pedro ; Maffei, Matteo: A^2L: Anonymous Atomic Locks for Scalability in Payment Channel Hubs. In: *2021 IEEE Symposium on Security and Privacy*. San Francisco, CA, USA : IEEE Computer Society Press, Mai 24–27, 2021, S. 1834–1851