Ring Signatures with Personalized Anonymity

Kyosuke Yamashita[1] and Keisuke Hara[2]

[1] Chuo University, Japan
[2] The University of Osaka, Japan
kyamashita887@g.chuo-u.ac.jp
hara-keisuke@ist.osaka-u.ac.jp

Abstract. Ring signatures have long struggled to balance absolute anonymity with traceability. While various extensions, such as traceable and accountable ring signatures, have been proposed, they typically apply a uniform anonymity or traceability rule to all potential signers. This paper introduces personalized-anonymity ring signatures (PARS), a novel primitive in which users are certified with different anonymity rights according to their roles or authority. Unlike ordinary ring signatures, PARS involves a group manager at key issuance; however, the group manager does not determine a fixed signing group. Instead, signers retain the ring-signature feature of choosing the ring at signing time, while the manager certifies whether each user's key permits fully anonymous signing or only traceable signing. This model is particularly suited for organizational governance, where ordinary members may require strong anonymity for internal reporting or expressing dissenting opinions, while users with institutional authority must remain accountable for official approvals or authorizations. We provide a formal syntax and rigorous security definitions for PARS, capturing both standard ring-signature requirements and traceability-related guarantees. We then present a generic construction from standard cryptographic primitives, including digital signatures, one-time signatures, public-key encryption, and non-interactive zero-knowledge proofs of knowledge.

1 Introduction

1.1 Background

The balance between privacy and traceability has long been a central theme in the design of privacy-preserving cryptographic primitives. Ring signatures [5,27] allow a signer to remain anonymous within a set of potential signers (the ring). While traditional ring signatures provide absolute anonymity, this feature can be abused for malicious purposes, leading to the development of various extensions aimed at introducing traceability in ring signatures.

Linkable ring signatures [20] allow anyone to determine if two signatures were produced by the same member, while traceable ring signatures [11] reveal the signer's identity only if they sign multiple messages using the same tag. Accountable ring signatures [6,31] introduce a tracer who can revoke anonymity

under specific conditions. However, these existing schemes generally treat all members of a group or ring as equal in terms of their privacy rights.

In many real-world organizations, such as companies, universities, and research institutes, anonymity needs are inherently asymmetrical and personalized. Ordinary members may need strong anonymity when submitting internal reports, expressing dissenting opinions, or participating in sensitive surveys. In contrast, members with institutional authority, such as executives, committee chairs, or budget holders, should remain accountable when issuing official approvals or authorizing actions on behalf of the organization.

This asymmetry is not directly captured by existing ring-signature variants, which typically provide anonymity or traceability according to a uniform rule applied to all potential signers. The desired functionality is to preserve the signer-chosen-ring feature of ordinary ring signatures while allowing the system to certify different anonymity rights for different users. Thus, ordinary members can remain anonymous even against the opening authority, whereas authority-holding members remain traceable when accountability is required. This motivates the following question:

How can we achieve a ring signature where the level of anonymity is personalized on a per-user basis?

1.2 Our Contribution

In this paper, we propose a new cryptographic primitive called *personalized-anonymity ring signature* (PARS) to solve the above problem. Our contributions are summarized as follows:

Conceptual Framework and Formalization. We introduce the formal syntax for PARS, which incorporates a multi-authority model to ensure both traceability and privacy. The proposed PARS scheme possesses the following characteristics:

Interactive Key Generation: Unlike traditional ring signatures where users generate keys locally, PARS utilizes an interactive protocol between a user and a group manager (GM). During this process, the GM evaluates the user's attributes (e.g., organizational rank) to determine whether the resulting signing key will be endowed with full anonymity or restricted to traceable signing. A distinctive feature of PARS is that this anonymity policy is fixed at key issuance, rather than being chosen ad hoc at signing time or determined only at opening time. Once the GM issues a certificate to a user, the certificate cryptographically binds the user's identity, public key, and anonymity permission. This makes the user's anonymity right a certified property of the signing key.

Separation of Duties: To enhance security and prevent the concentration of power, we decouple the administrative roles into two distinct entities: the GM, who handles membership and key issuance, and the opening authority (OA), a trusted third party responsible for de-anonymizing signatures when required.

Rigorous Security Definitions: We establish a comprehensive security model for PARS. This includes standard ring signature requirements such as unforgeability and anonymity, augmented with properties typically found in group signatures: (i) traceability ensures that any valid signature produced by a user with a "traceable" key with respect to a ring can be linked to the identity of one of the ring members by the OA, and (ii) non-frameability protects users from being falsely accused of generating a signature, ensuring that neither the GM, OA, nor other users can forge a signature that attributes back to an innocent member.

Leakage Deterrence via Dual-Capability Keys: One subtle issue in PARS is preventing the unauthorized transfer of "anonymous" keys. If a signing key only allows for anonymous signing, a user might be tempted to share it with unauthorized third parties. To counter this, we design the PARS signing key such that even those granted anonymity retain the technical ability to produce traceable signatures. This causes a "risk-of-exposure" incentive: since the key can generate signatures tied to the owner's identity, the owner is deterred from leaking the key to others, fearing that the recipient might generate traceable (and thus damaging) signatures in the owner's name.

Generic Construction from Standard Primitives. We provide a generic construction of PARS utilizing standard cryptographic building blocks: digital signatures, one-time signatures, public-key encryption (PKE) satisfying indistinguishability under adaptive chosen-ciphertext attacks (IND-CCA2 security), and simulation-sound non-interactive zero-knowledge proofs of knowledge (NIZKPoK). One of the primary advantages of our PARS scheme is that, despite introducing a novel conceptual framework, it can be fully instantiated using standard cryptographic primitives. This ensures that the security of our construction rests on well-established foundations rather than ad-hoc assumptions. (We only discuss such instantiations in Section 1.3.)

In the following, we provide an overview of our construction, focusing on the setup, key generation, and signing algorithms. Note that one-time signatures are omitted here to simplify the explanation.

Setup Algorithm: The setup algorithm first chooses a binary function f, which takes a user's identity as input and outputs 1 if anonymous signing is permitted for that user, and 0 otherwise. Next, it generates the key pairs for GM and OA. The GM's keys are generated via a standard digital signature key generation algorithm, where the signing key serves as the master secret key. The OA's keys are produced using a key generation algorithm of PKE, with the decryption key acting as the OA's secret key. Finally, the algorithm generates a common reference string (CRS) of the NIZKPoK and outputs the public parameters, which include f, the CRS, the verification key, and the public key.

We emphasize that our construction not only allows for switching between anonymity and traceability but also offers greater personalization. Let f be a function whose range is $[I] = \{1, 2, \ldots, I\}$ for a given integer I so that $f(\mathsf{id}) = i$

means the user id's attribute is i. By adapting the language of the NIZKPoK to be compatible with this function, we can more finely configure the user information to be disclosed. For a more detailed discussion, see Section 4.3.

Key-Generation Algorithm: As previously stated, the key generation is an interactive algorithm between the GM and the user. The user first locally generates a standard signature key pair and sends the verification key vk and their identity id to the GM. Using the master secret key, the GM generates a signature on the message $(\mathsf{id}, \mathsf{vk}, f(\mathsf{id}))$. Note that if $f(\mathsf{id}) = 1$, then the user is permitted to create an anonymous signature. The GM then transmits this signature to the user as a certificate. The user's secret key consists of the standard signing key and the certificate, while the public key is vk.

Signing Algorithm: In our PARS construction, a signature primarily consists of a NIZKPoK proof and a ciphertext. The ciphertext is formed by encrypting the signer's id under the public key of the OA. Consequently, the OA traces the signer by decrypting this ciphertext. However, if $f(\mathsf{id}) = 1$, the signer is permitted to encrypt 0 instead of their actual id. In this case, the signature becomes untraceable even for the OA.

The language proven by the NIZKPoK plays an essential role in the signature. With respect to a set of user public keys (i.e., the ring R), the NIZKPoK demonstrates the following knowledge:

- Knowledge regarding a certificate of a user included in R.
- Knowledge regarding the plaintext of the ciphertext; specifically, that the ciphertext has been constructed based on the correct plaintext in accordance with the value of $f(\mathsf{id})$.

The first condition guarantees that one of the ring members produced the signature. Furthermore, due to the zero-knowledge property of the NIZKPoK (and the indistinguishability of the PKE), it is impossible for any party other than the OA to identify the actual signer. See Section 4.3 for a more precise description of the language.

Comparison with Existing Ring Signatures: Table 1 summarizes the comparison between PARS and existing ring signatures. The most distinctive functional feature of PARS, compared with existing ring signatures, is that anonymity is personalized on a per-user basis. In ordinary ring signatures, anonymity is a uniform property provided with respect to a signer-chosen ring. In contrast, PARS allows the system to specify, at key-issuance time, whether each user is authorized to generate anonymous signatures. This functionality is enabled by introducing a GM that controls users' signing capabilities, while retaining the ring-signature flavor that signatures are verified with respect to a set of public keys. Thus, the role of the GM in PARS is not to define a fixed group of anonymous signers as in group signatures, but to personalize anonymity permissions among users who may later participate in signer-chosen rings.

Table 1. Comparison between PARS and existing ring signatures. "RS" stands for ring signatures.

Primitive	Anonymity policy	Key generation	Dual-capability key	Accountability mechanism
Ordinary RS [5,27]	Uniform	Locally	–	–
Linkable RS [20]	Uniform	Locally	–	Public linkability
Traceable RS [11]	Uniform	Locally	–	Public tracing
Accountable RS [6,31]	Uniform	Locally	–	Opener
PARS	Personalized	Interaction with GM	✓	OA

1.3 Discussion on Concrete Instantiations

The primary goal of this paper is to introduce the concept of PARS and to present a generic construction of it. Nevertheless, we also discuss concrete instantiations here. Recall that our PARS is realized from standard signature, OTS, IND-CCA2 PKE, and simulation-sound NIZKPoK. We focus on the IND-CCA2 PKE and simulation-sound NIZKPoK components, since they are the main sources of concrete assumptions. The standard signature and one-time signature components can be instantiated from standard one-way function-based constructions.

Instantiation under SXDH. Under SXDH-type assumptions in asymmetric bilinear groups, the IND-CCA2 PKE component can be instantiated by a DDH-based CCA-secure encryption scheme, such as Cramer–Shoup [9]. For the proof component, we can rely on Groth's simulation-sound NIZKPoK techniques in bilinear groups [13]. Hence, together with standard signatures and one-time signatures, our construction yields a PARS scheme under SXDH-type assumptions.

Instantiation under LWE. Under the LWE assumption, the IND-CCA2 PKE component can be instantiated using lattice-based CCA-secure encryption techniques, for example via lossy trapdoor functions [26]. Moreover, as discussed in [7, Fact 3.2], Peikert and Shiehian [25] give NIZK for NP from plain LWE, and generic compilers such as proposed in [10] can upgrade NIZK for NP, together with PKE and OTS, to simulation-sound extractable NIZKPoK for NP. This gives a lattice-based route to the NIZKPoK component required in our construction. Consequently, PARS can also be instantiated from the LWE assumption, up to the use of these known generic transformations.

1.4 Related Work

Group signatures [3,4,8] represent another class of privacy-preserving signature schemes. Similar to ring signatures, a group signature is a group-oriented signature scheme; however, it typically differs from the former in the following

respects: (i) The OA and the GM are present, with the OA (unconditionally) capable of tracing all users. (ii) The group of potential signers is determined by the GM. It has been frequently pointed out that the traceability of group signatures is excessively powerful, leading to numerous studies aimed at mitigating this authority: tracing-by-linking [14,30], traceable signatures [15], message-dependent opening [28], decentralized-opening [22], and threshold tracing [12].

These variants restrict how, when, or by whom opening is performed, while applying a uniform tracing policy to all group members. A notable exception is accountable tracing signatures (ATS) [16,19,29], where the GM determines at enrollment whether each user's signatures are traceable or perfectly untraceable. Despite this conceptual similarity, ATS differs from PARS in several respects. First, ATS hides the tracing status even from the user, whereas in PARS the anonymity permission is publicly determined by f and bound to the signing key by the GM's certificate. Second, ATS realizes untraceability through an escrow key whose secret key is unknown, thereby fixing the signing mode at enrollment. In contrast, PARS allows users with $f(\mathsf{id}) = 1$ to choose between anonymous and traceable signing, enabling our dual-capability mechanism against key leakage. Finally, ATS is a group-signature primitive with an authority-managed signer set and a single key-issuing/opening authority, whereas PARS preserves signer-chosen rings and separates the GM from the OA.

Other primitives conceptually related to PARS include bifurcated anonymous signatures (BiAS) [18] and multimodal private signatures (MPS) [23]. BiAS allows a signer to choose whether a signature is traceable at the time of its creation. A key feature of BiAS is that traceability is determined according to the content of the message. Multimodal private signatures allow for the granular setting of information regarding the signer that can be disclosed by the OA. Since both are categorized as group signatures, PARS is considered capable of supporting more flexible applications. In particular, for applications that require the flexible selection of a signer group, such as whistleblowing, PARS is more suitable than these existing schemes. However, as PARS currently does not support message-dependent traceability, we leave these as subjects for future work.

Attribute-based signatures [21] are a type of anonymous signatures in which verification requires only information about the signer's attributes rather than their identity. As discussed in Section 1.2, by following our framework, it is possible to construct a scheme in which the OA reveals only the attributes of the signer. Policy-based signatures [1] are signature schemes in which a signing key is associated with a predicate P, and the signer is only allowed to sign a message m if and only if $P(\mathsf{m}) = 1$.

There are ring signatures that allow a signer to prove whether they did or did not create a specific public signature in previous literature [17,24]. These schemes differ from PARS in that the signers themselves voluntarily disclose the facts regarding their involvement in the signature generation. Nevertheless, they share a common objective with PARS in addressing the challenge of balancing anonymity with traceability.

2 Preliminaries

Notation. We denote by $\lambda \in \mathbb{N}$ a security parameter throughout this paper. For $n \in \mathbb{N}$, we let $[n]$ be the set $\{1, 2, \cdots, n\}$. A polynomial function and a negligible function are denoted by $\mathsf{poly}()$ and $\mathsf{negl}()$, respectively. Probabilistic polynomial-time is abbreviated as PPT. For an NP language $\mathcal{L}$, the NP relation is denoted by $R_{\mathcal{L}}$. If an algorithm X has a subroutine A, we denote it by $X.A$. We let $\mathcal{ID}$ be an identifier space of users. In particular, we assume that $0 \notin \mathcal{ID}$. We treat 0 as a special symbol representing an anonymous signature. An arbitrary value is denoted by a dot $\cdot$.

In what follows, we define cryptographic primitives that will be used in our generic construction.

Definition 1 (Signature). *A signature scheme consists of three PPT algorithms* $(\mathsf{KG}, \mathsf{Sig}, \mathsf{Vrf})$ *that work as follows:*

$\mathsf{KG}(1^\lambda) \to (\mathsf{sk}, \mathsf{vk})$: *The key generation algorithm, given a security parameter* 1^λ*, outputs a secret key* sk *and a verification key* vk*.*

$\mathsf{Sig}(\mathsf{sk}, \mathsf{m}) \to \sigma$: *The signing algorithm, given a secret key* sk *and a message* m*, outputs a signature* σ*.*

$\mathsf{Vrf}(\mathsf{vk}, \mathsf{m}, \sigma) = 1/0$: *The (deterministic) verification algorithm, given a verification key* vk*, a message* m*, and a signature* σ*, outputs* 1 *(meaning "valid") or* 0 *(meaning "invalid").*

Definition 2 (EUF-CMA). *A signature scheme* $\Pi = (\mathsf{KG}, \mathsf{Sig}, \mathsf{Vrf})$ *is existentially unforgeable under an adaptive chosen-message attack (EUF-CMA) if for any sufficiently large security parameter* λ*, and any PPT adversary* $\mathcal{A}$*, it holds that* $\Pr[\mathsf{ExpEUFSig}_{\Pi,\mathcal{A}}(1^\lambda) = 1] \leq \mathsf{negl}(\lambda)$*, where* $\mathsf{ExpEUFSig}$ *is defined as follows:*

$$\begin{array}{l} \underline{\mathsf{ExpEUFSig}_{\Pi,\mathcal{A}}(1^\lambda)} \\ L := \emptyset; (\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KG}(1^\lambda); \\ (\mathsf{m}^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{O}_{\mathsf{Sig}}}(\mathsf{vk}) : \\ \text{Output 1 if } \mathsf{Vrf}(\mathsf{vk}, \mathsf{m}^*, \sigma^*) = 1 \wedge \mathsf{m}^* \notin L, \text{otherwise 0} \end{array}$$

where $\mathsf{O}_{\mathsf{Sig}}$ *works as follows: Given a message* m*, it returns* $\sigma \leftarrow \mathsf{Sig}(\mathsf{sk}, \mathsf{m})$*, and updates* $L := L \cup \{\mathsf{m}\}$*.*

Note that one-time signatures (OTS) follow the same syntax, but we denote the secret key and verification key by osk and ovk, respectively. The security requirement for OTS is defined as follows.

Definition 3 (sOT-EUF-CMA). *An OTS* $\Pi = (\mathsf{KG}, \mathsf{Sig}, \mathsf{Vrf})$ *is strong existentially unforgeable under an adaptive chosen-message attack (sOT-EUF-CMA) if for any sufficiently large security parameter* λ*, and any PPT adversary* $\mathcal{A}$*, it holds that* $\Pr[\mathsf{ExpEUFOTS}_{\Pi,\mathcal{A}}(1^\lambda) = 1] \leq \mathsf{negl}(\lambda)$*, where* $\mathsf{ExpEUFOTS}$ *is defined as follows:*

$$\begin{array}{l} \underline{\mathsf{ExpEUFOTS}_{\Pi,\mathcal{A}}(1^\lambda)} \\ (\mathsf{osk}, \mathsf{ovk}) \leftarrow \mathsf{KG}(1^\lambda); \mathsf{m} \leftarrow \mathcal{A}(\mathsf{ovk}); \sigma \leftarrow \mathsf{Sig}(\mathsf{osk}, \mathsf{m}); (\mathsf{m}^*, \sigma^*) \leftarrow \mathcal{A}(\sigma) : \\ \text{Output 1 if } \mathsf{Vrf}(\mathsf{ovk}, \mathsf{m}^*, \sigma^*) = 1 \wedge (\mathsf{m}, \sigma) \neq (\mathsf{m}^*, \sigma^*), \text{otherwise 0} \end{array}$$

Definition 4 (Public Key Encryption). *A public key encryption (PKE) scheme consists of the following three PPT algorithms* $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$*:*

$\mathsf{KG}(1^\lambda) \to (\mathsf{sk}, \mathsf{pk})$**:** *Given* 1^λ*, output a secret key* sk *and a public key* pk*.*
$\mathsf{Enc}(\mathsf{pk}, m; r) \to c$**:** *Given a public key* pk*, a plaintext* m*, and a randomness* r*, output a ciphertext* c*.*
$\mathsf{Dec}(\mathsf{sk}, c) = m/\bot$**:** *Given a secret key* sk *and a ciphertext* c*, output a plaintext* m *or* $\bot$*, where* $\bot$ *indicates* c *is invalid.*

Definition 5 (IND-CCA2 Security). *A PKE scheme* $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ *is indistinguishable under adaptive chosen ciphertext attack (IND-CCA2), if for any security parameter* λ *and any stateful PPT adversary* $\mathcal{A}$*, it holds that* $|\Pr[\mathsf{ExpPKE}_{\mathcal{A},\Pi}(1^\lambda) = 1] - 1/2| \leq \mathsf{negl}(\lambda)$ *where* ExpPKE *is defined as follows:*

$\mathsf{ExpPKE}_{\mathcal{A},\Pi}(1^\lambda)$
$L := \emptyset; (\mathsf{sk}, \mathsf{pk}) \leftarrow \Pi.\mathsf{KG}(\lambda);$
$(m_0, m_1) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pk});$
$b \leftarrow \{0,1\}; c^* \leftarrow \Pi.\mathsf{Enc}(\mathsf{pk}, m_b);$
$b' \leftarrow \mathcal{A}^{\mathcal{O}}(c^*)$:
Output 1 *if* $b' = b$, otherwise 0

The oracle $\mathcal{O}$ *is that, given a ciphertext* c*, returns* $\mathsf{Dec}(\mathsf{sk}, c)$ *and updates* $L := L \cup \{c\}$*. Further, if* c^* *is given after the challenger creates it, then* $\mathcal{O}$ *returns* $\bot$*.*

We define non-interactive zero-knowledge proof (NIZK) and non-interactive zero-knowledge proof of knowledge (NIZKPoK). The formalization follows [13].

Definition 6 (NIZK). *Let* $\mathcal{L}$ *be an NP-language. Let* $\Pi = (\mathsf{CRS}, \mathsf{Prv}, \mathsf{Vrf})$ *be a tuple of PPT algorithms that work as follows.*

$\mathsf{CRS}(1^\lambda) \to \mathsf{crs}$**:** *Given* 1^λ*, output a common reference string (CRS)* crs*.*
$\mathsf{Prv}(\mathsf{crs}, x, w) \to \pi$**:** *Given a CRS* crs *and a pair of a statement and a witness* $(x, w) \in R_{\mathcal{L}}$*, output a proof* π*.*
$\mathsf{Vrf}(\mathsf{crs}, x, \pi) \to 1/0$**:** *Given a CRS* crs*, a statement* x*, and a proof* π*, output a bit* $b \in \{0,1\}$*, where* 1 *means "accept" and* 0 *means "reject."*

The tuple Π *is a non-interactive zero-knowledge proof system (NIZK) for* $\mathcal{L}$ *if it is complete, sound, and zero-knowledge, defined as follows:*

Completeness. *For any* λ*, any* $\mathsf{crs} \leftarrow \Pi.\mathsf{CRS}(1^\lambda)$ *and any* $(x, w) \in R_{\mathcal{L}}$*, it holds that* $\Pr[\Pi.\mathsf{Vrf}(\mathsf{crs}, x, \Pi.\mathsf{Prv}(\mathsf{crs}, x, w)) = 1] \geq 1 - \mathsf{negl}(\lambda)$*.*
Soundness. *For any PPT adversary* $\mathcal{A}$*, it holds that* $\Pr[\mathsf{crs} \leftarrow \Pi.\mathsf{CRS}(1^\lambda), (x, \pi) \leftarrow \mathcal{A}(\mathsf{crs}) : \Pi.\mathsf{Vrf}(\mathsf{crs}, x, \pi) = 1 \wedge x \notin \mathcal{L}] \leq \mathsf{negl}(\lambda)$*.*
(Adaptive) Zero-Knowledge. *There exist PPT algorithms* SimCRS *and* Sim *that work as follows.*
$\mathsf{SimCRS}(1^\lambda) \to (\mathsf{crs}, \tau)$**:** *Given* 1^λ*, output a CRS* crs *and a trapdoor* τ*.*
$\mathsf{Sim}(\mathsf{crs}, x, \tau)$**:** *Given a CRS* crs*, a statement* $x \in \mathcal{L}$ *and a trapdoor* τ*, output a simulated proof* π*.*

Further, for any λ and any PPT adversary $\mathcal{A}$, it holds that

$$\begin{aligned}&|\Pr[\mathsf{crs} \leftarrow \Pi.\mathsf{CRS}(\lambda) : 1 \leftarrow \mathcal{A}^{\mathcal{O}_0(\cdot,\cdot)}(\sigma)]\\ &-\Pr[(\mathsf{crs},\tau) \leftarrow \Pi.\mathsf{SimCRS}(\lambda) : 1 \leftarrow \mathcal{A}^{\mathcal{O}_1(\cdot,\cdot)}(\mathsf{crs})]| \quad \leq \mathsf{negl}(\lambda)\end{aligned}$$

where $\mathcal{O}_0$ and $\mathcal{O}_1$ are oracles that, given (x,w), output $\perp$ if $(x,w) \notin R_{\mathcal{L}}$, otherwise they output $\Pi.\mathsf{Prv}(\mathsf{crs},x,w)$ and $\Pi.\mathsf{Sim}(\mathsf{crs},x,\tau)$, respectively.

We note that the simulated CRS is computationally indistinguishable from the honestly generated CRS.

Definition 7 (NIZKPoK). *Let Π be an NIZK for an NP-language $\mathcal{L}$. It is a non-interactive zero-knowledge proof of knowledge (NIZKPoK) if Π is simulation-sound extractable, as defined below:*

Simulation-Sound Extractability. *Let* ExtCRS *and* Ext *be polynomial-time algorithms that work as follows:*

$\mathsf{ExtCRS}(1^\lambda) \to (\mathsf{crs},\tau,\zeta)$**:** *Given 1^λ, output a CRS* crs*, a trapdoor τ, and a trapdoor ζ for extraction.*

$\mathsf{Ext}(\mathsf{crs},\zeta,x,\pi) \to w/\perp$**:** *Given a CRS* crs*, a trapdoor ζ for extraction, a statement x, and a proof π, output a witness w or $\perp$.*

Let Q be an initially empty set. The algorithm Π is simulation-sound extractable if there is a pair of polynomial-time algorithms $(\mathsf{ExtCRS},\mathsf{Ext})$ *s.t. for any λ and PPT adversary $\mathcal{A}$, it holds that*

$$\begin{aligned}\Pr[(\mathsf{crs},\tau,\zeta) \leftarrow \Pi.\mathsf{ExtCRS}(1^\lambda),(x,\pi) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{crs},\zeta),w \leftarrow \Pi.\mathsf{Ext}(\mathsf{crs},\zeta,x,\pi) :\\ (x,\pi) \notin Q \wedge (x,w) \notin R_{\mathcal{L}} \wedge \Pi.\mathsf{Vrf}(\mathsf{crs},x,\pi) = 1] \leq \mathsf{negl}(\lambda)\end{aligned}$$

where $\mathcal{O}$ is an oracle that, given (x,w), outputs $\pi \leftarrow \Pi.\mathsf{Sim}(\mathsf{crs},x,\tau)$ and updates $Q := Q \cup \{(x,\pi)\}$ if $(x,w) \in R_{\mathcal{L}}$, otherwise $\perp$.

We note that the CRS for extraction is computationally indistinguishable from the honestly generated CRS.

3 Definition of Personalized-Anonymity Ring Signature

3.1 Syntax

As discussed in Section 1.2, we consider the opening authority (OA) in addition to the group manager (GM) to enable a more fine-grained security analysis. Furthermore, the key generation procedure is executed as an interactive protocol between the GM and the users. During this interaction, the GM determines whether the user is permitted to generate anonymous signatures and issues a corresponding certificate.

Let f be a binary function that takes $\mathsf{id} \in \mathcal{ID}$ as input. This function is used by the GM to make the following decision: if $f(\mathsf{id}) = 1$, the user is allowed to generate anonymous signatures; otherwise, the signatures must be traceable.

Furthermore, even if $f(\mathsf{id}) = 1$, we allow the corresponding signer to create a signature in a traceable manner. We argue that this is necessary to avoid the following attack. If we adopt a setting in which a signing key with $f(\mathsf{id}) = 1$ can be used only to generate anonymous signatures, then there is a risk that the signer may leak (or transfer) their signing key to others. Since this key can produce only anonymous signatures, the signer can completely conceal whose key was used to generate a signature without incurring any risk. Therefore, allowing a secret key corresponding to $f(\mathsf{id}) = 1$ to generate traceable signatures as well prevents such attacks.

It should be emphasized that defining PARS for more general functions allows for greater flexibility in personalization. However, since distinguishing between users permitted to generate anonymous signatures and those subject to traceability is particularly significant from both theoretical and practical perspectives, this paper focuses on the aforementioned binary functions.

The syntax of PARS is provided as follows.

Definition 8. *Let $\mathcal{F}$ be a binary function family. A personalized-anonymity ring signature (PARS) scheme (with respect to $\mathcal{F}$) consists of the following polynomial-time algorithms.*

$\mathsf{Set}(\mathcal{F}) \to (\mathsf{pp}, \mathsf{msk}, \mathsf{oak})$**:** *The setup algorithm takes a binary function family $\mathcal{F}$ as an input, and outputs a public parameter* pp*, a master secret key* msk*, and an opening authority's key* oak*. We require that a binary function f is chosen from $\mathcal{F}$ in this procedure, and is contained in* pp*.*

$\langle \mathsf{GKG}(\mathsf{msk}, \mathsf{reg}), \mathsf{UKG}(\mathsf{id}) \rangle (\mathsf{pp}) \to (\mathsf{usk}, \mathsf{upk})$**:** *The key generation algorithm is an interactive algorithm between the GM (*GKG*) and a user (*UKG*). Let* trans *be the transcript exchanged between* GKG *and* UKG*. When the interaction is completed successfully,* GKG *updates* $\mathsf{reg} := \mathsf{reg} \cup \{\mathsf{trans}\}$ *and* UKG *obtains the user's secret/public key* $(\mathsf{usk}, \mathsf{upk})$*. We require that* GKG *gives* UKG *a certificate* $\mathsf{cert}_{\mathsf{id}}$ *that determines if the user* id *can create a signature in an anonymous or traceable manner. Further,* upk *contains* id*, and we assume that* id *can be extracted from* upk *whenever necessary. We sometimes denote the public key of a user* id *by* $\mathsf{upk}_{\mathsf{id}}$*.*

$\mathsf{RSig}^{\mathsf{A}}(\mathsf{pp}, \mathsf{usk}, \mathsf{R}, \mathsf{m}, b) \to \sigma$**:** *This algorithm is run by a signer who is allowed to create a signature in an anonymous manner. Given a public parameter* pp*, a signer's secret key* usk*, a ring* $\mathsf{R} = (\mathsf{upk}_i)_{i \in [n]}$ *where* $n = \mathsf{poly}(\lambda)$*, a message* m*, and a bit* $b \in \{0, 1\}$ *that determines anonymous (*$b = 1$*)/traceable (*$b = 0$*) mode, it outputs a signature* σ*.*

$\mathsf{RSig}^{\mathsf{T}}(\mathsf{pp}, \mathsf{usk}, \mathsf{R}, \mathsf{m}) \to \sigma$**:** *This algorithm is run by a signer who is* not *allowed to create a signature in an anonymous manner. Given a public parameter* pp*, a signer's secret key* usk*, a ring* $\mathsf{R} = (\mathsf{upk}_i)_{i \in [n]}$ *where* $n = \mathsf{poly}(\lambda)$*, and a message* m*, it outputs a signature* σ*.*

$\mathsf{Vrf}(\mathsf{pp}, \mathsf{R}, \mathsf{m}, \sigma) = 1/0$**:** *Given a public parameter* pp*, a ring* R*, a message* m*, and a signature* σ*, it outputs* 1 *or* 0*.*

$\mathsf{Open}(\mathsf{pp}, \mathsf{oak}, \mathsf{R}, \mathsf{m}, \sigma) \to \mathsf{id}/\perp$**:** *Given a public parameter* pp*, an opening authority's key* oak*, a ring* R*, a message* m*, and a signature* σ*, it outputs a user's identifier* id *or* $\perp$*.*

3.2 Security Requirements

We formally define security requirements for PARS in this section. Before providing the rigorous description, we provide an intuitive explanation of the security formulation and its necessity, followed by the description of oracles that will be used in security games. We note that we do not define security against the unauthorized transfer of anonymous keys that is discussed in Section 1, because it is trivially prevented by the dual-capability keys.

In addition to standard security requirements for ring signatures, i.e., unforgeability and anonymity, we need to define traceability-related security properties for PARS, given its integration of the tracing functionality. In the following, we provide an intuitive explanation of these properties.

Traceability-related security properties. We introduce three security requirements for this type of property. The first one is non-frameability, which means that malicious signers cannot produce a signature that can be opened to an honest signer in the ring. In particular, we require this security property to hold even against a malicious GM and OA.

The second requirement is traceability, which means that a signature created using a traceable signing key should be traced. In this requirement, we allow the adversary to corrupt OA, but not the GM. This is because, if the GM were corrupted, it would become possible to issue anonymous keys to identifiers for which anonymous signing should not be permitted.

The third requirement is "traceability soundness." This requires that, for a valid signature with respect to a ring, its opening result should not be outside of the ring. While the term opening soundness is typically used to ensure the validity of the evidence provided by an opener, we use traceability soundness to specifically denote that any valid signature must trace back to an identity within the designated ring.

Unforgeability. We follow the standard definition of unforgeability for ring signatures. That is, it is hard to create a valid signature with respect to a ring without a secret key of a ring member. While it is well-known that traceability and anonymity (and non-frameability in dynamic group signatures) imply unforgeability in group signatures, we emphasize that for PARS, unforgeability must be explicitly defined.

In group signatures, the group is fixed and managed by the GM. Hence, the result of the opening algorithm is expected to be one of the registered group members generated through the setup/join procedure. This makes traceability and anonymity (and non-frameability) closely related to authenticity: if a valid signature cannot be properly opened, traceability is violated; if it opens to an honest user who did not sign, non-frameability is violated.

In PARS, the situation is different because the ring is chosen by the signer. Although the opening result may correspond to an identity certified by the GM, this identity is not necessarily contained in the ring claimed by the signature. Thus, there is a possible gap between "the signature opens to some certified

identity" and "the signature was generated using a signing key of a member of the claimed ring." For example, an adversary may output a valid signature with respect to a ring consisting of honest users, while the opening result is a malicious identity outside of the ring. Such an attack does not frame any honest ring member and does not necessarily violate traceability, but it is still a forgery as a ring signature. Thus, in PARS, unforgeability must be required explicitly.

As will become clear later, the traceability-related security properties are not sufficient to imply unforgeability. We argue that this arises from the different notions of a "group" in PARS and group signatures. That is, in group signatures, a group is entirely defined by the GM. On the other hand, in PARS, signers are allowed to choose groups (like ring signatures).

Anonymity-related security properties. Anonymity also requires careful consideration, specifically regarding whether a signature is generated with an anonymous key, and whether the OA is corrupted by the adversary. Consequently, we define two types of anonymity. The first ensures that if the OA is not corrupted, the signer remains unidentified regardless of which key was used (OA-honest anonymity). The second guarantees that a signature produced with an anonymous key remains anonymous even if the adversary corrupts the OA (OA-corrupted anonymity). Although OA-corrupted anonymity seems stronger than OA-honest anonymity, they are incomparable. This is due to the fact that the types of keys available to the adversary vary among these security definitions.

Although we consider a malicious GM in our formalizations, we require that the setup algorithm is run honestly in the experiment. We might be able to introduce a malicious setup in our definition, and we will discuss this point in Section 4.3. We briefly explain the reason for this modeling choice. Recall that we employ a NIZKPoK in our construction of PARS. Allowing a malicious setup would, in particular, allow the CRS of the NIZKPoK to be generated maliciously. Since our anonymity proof relies on the zero-knowledge property of the underlying NIZKPoK, such a subverted CRS may invalidate the argument by embedding hidden information that enables the setup authority to know the witness used in a proof. Achieving anonymity in this stronger model would therefore require additional non-standard techniques. In contrast, one of the main goals of this paper is to demonstrate that the core functionality of PARS can be obtained from standard cryptographic primitives. We therefore adopt the honest-setup model in our formalization and regard malicious-setup anonymity as an important extension to be addressed in future work.

Oracles. Here, we describe the oracles that are used in the experiments. We assume that the parameters $\mathsf{pp}, \mathsf{msk}$ and oak are created by the challenger in the experiments, and the lists $\mathsf{reg}, \mathsf{L}_{\mathsf{KG}}, \mathsf{L}_{\mathsf{SK}}, \mathsf{L}_{\mathsf{HU}}, \mathsf{L}_{\mathsf{CU}}, \mathsf{L}^{\mathsf{A}}_{\mathsf{Sig}}$, and $\mathsf{L}^{\mathsf{T}}_{\mathsf{Sig}}$ are initialized as empty sets. Further, we assume without loss of generality that a ring is denoted by $\mathsf{R} = \{\mathsf{upk}_{\mathsf{id}_i}\}_{i\in[n]}$ where $n = \mathsf{poly}(\lambda)$.

$\mathsf{O}_{\mathsf{KG}}(\mathsf{id})$**:** Given an identifier id, if $(\mathsf{id}, \cdot) \in \mathsf{L}_{\mathsf{KG}} \cup \mathsf{L}_{\mathsf{HU}} \cup \mathsf{L}_{\mathsf{CU}}$, then return $\perp$.

Otherwise, this oracle runs $\langle \mathsf{GKG}(\mathsf{msk}, \mathsf{reg}), \mathsf{UKG}(\mathsf{id})\rangle(\mathsf{pp}) \to (\mathsf{usk}, \mathsf{upk})$. Let

trans be the transcript that is obtained during the process. It outputs upk, and updates $\mathsf{reg} := \mathsf{reg} \cup \{\mathsf{trans}\}, \mathsf{L}_{\mathsf{KG}} := \mathsf{L}_{\mathsf{KG}} \cup \{\mathsf{upk}\}$ and $\mathsf{L}_{\mathsf{SK}} := \mathsf{L}_{\mathsf{SK}} \cup \{\mathsf{id}, \mathsf{usk}\}$. Recall that id can be extracted from upk.

$\mathsf{O}_{\mathsf{UKG}}(\mathsf{id})$**:** Given an identifier id, if $(\mathsf{id}, \cdot) \in \mathsf{L}_{\mathsf{KG}} \cup \mathsf{L}_{\mathsf{HU}} \cup \mathsf{L}_{\mathsf{CU}}$, then return $\perp$. This oracle plays the role of $\mathsf{UKG}(\mathsf{id})$, while the adversary plays the role of (corrupted) $\mathsf{GKG}(\mathsf{msk}, \mathsf{reg})$, under the public parameter pp. When the interaction is completed successfully, the oracle updates $\mathsf{reg} := \mathsf{reg} \cup \{\mathsf{trans}\}, \mathsf{L}_{\mathsf{SK}} := \mathsf{L}_{\mathsf{SK}} \cup \{\mathsf{id}, \mathsf{usk}\}$ and $\mathsf{L}_{\mathsf{HU}} := \mathsf{L}_{\mathsf{HU}} \cup \{\mathsf{upk}\}$.

$\mathsf{O}_{\mathsf{GKG}}$**:** This oracle plays the role of $\mathsf{GKG}(\mathsf{msk}, \mathsf{reg})$, while the adversary plays the role of $\mathsf{UKG}(\mathsf{id})$ for $(\mathsf{id}, \cdot) \notin \mathsf{L}_{\mathsf{KG}} \cup \mathsf{L}_{\mathsf{HU}} \cup \mathsf{L}_{\mathsf{CU}}$, under a public parameter pp. Thus, if $\mathsf{O}_{\mathsf{GKG}}$ receives id s.t. $(\mathsf{id}, \cdot) \in \mathsf{L}_{\mathsf{KG}} \cup \mathsf{L}_{\mathsf{HU}} \cup \mathsf{L}_{\mathsf{CU}}$ from the adversary, it outputs $\perp$. When the interaction is completed successfully, the oracle updates $\mathsf{reg} := \mathsf{reg} \cup \{\mathsf{trans}\}$ and $\mathsf{L}_{\mathsf{CU}} := \mathsf{L}_{\mathsf{CU}} \cup \{\mathsf{upk}\}$.

$\mathsf{O}^{\mathsf{A}}_{\mathsf{RSig}}(\mathsf{upk}, \mathsf{R}, \mathsf{m}, b)$**:** Given a public key upk, a ring R, a message m, and a bit $b \in \{0, 1\}$, if $(\mathsf{upk}, \mathsf{R}, \mathsf{m}, b, \sigma) \in \mathsf{L}^{\mathsf{A}}_{\mathsf{Sig}}$, then return σ. If one of the following conditions holds, then return $\perp$: $(\mathsf{usk}, \mathsf{upk})$ cannot be retrieved from reg and L_{SK}, $f(\mathsf{id}) \neq 1$ for an id that can be extracted from upk, or $\mathsf{upk} \notin \mathsf{R}$. Otherwise, it outputs $\sigma \leftarrow \mathsf{RSig}^{\mathsf{A}}(\mathsf{pp}, \mathsf{usk}, \mathsf{R}, \mathsf{m}, b)$, and updates $\mathsf{L}^{\mathsf{A}}_{\mathsf{Sig}} := \mathsf{L}^{\mathsf{A}}_{\mathsf{Sig}} \cup \{(\mathsf{upk}, \mathsf{R}, \mathsf{m}, b, \sigma)\}$.

$\mathsf{O}^{\mathsf{T}}_{\mathsf{RSig}}(\mathsf{upk}, \mathsf{R}, \mathsf{m})$**:** Given a public key upk, a ring R, and a message m, if $(\mathsf{upk}, \mathsf{R}, \mathsf{m}, \sigma) \in \mathsf{L}^{\mathsf{T}}_{\mathsf{Sig}}$, then return σ. If one of the following conditions holds, then return $\perp$: $(\mathsf{usk}, \mathsf{upk})$ cannot be retrieved from reg and L_{SK}, $f(\mathsf{id}) \neq 0$ for an id that can be extracted from upk, or $\mathsf{upk} \notin \mathsf{R}$. Otherwise, it outputs $\sigma \leftarrow \mathsf{RSig}^{\mathsf{T}}(\mathsf{pp}, \mathsf{usk}, \mathsf{R}, \mathsf{m})$, and updates $\mathsf{L}^{\mathsf{T}}_{\mathsf{Sig}} := \mathsf{L}^{\mathsf{T}}_{\mathsf{Sig}} \cup \{(\mathsf{upk}, \mathsf{R}, \mathsf{m}, \sigma)\}$.

$\mathsf{O}_{\mathsf{Op}}(\mathsf{R}, \mathsf{m}, \sigma)$**:** Given a ring R, a message m, and a signature σ, output $\mathsf{Open}(\mathsf{pp}, \mathsf{oak}, \mathsf{R}, \mathsf{m}, \sigma)$ if $\mathsf{Vrf}(\mathsf{pp}, \mathsf{R}, \mathsf{m}, \sigma) = 1$. Otherwise, return $\perp$.

We let $\mathsf{O}^{\mathsf{T}}_{\mathsf{GKG}}$ be the key generation oracle that returns $\perp$ for id s.t. $f(\mathsf{id}) = 1$, but other behavior is the same as $\mathsf{O}_{\mathsf{GKG}}$. This oracle is used in the experiment for traceability. Moreover, $\mathsf{O}^{\sigma^*}_{\mathsf{Op}}$ is an oracle which returns $\perp$ if the signature σ^* is given, otherwise it behaves in the same way as O_{Op}. This oracle will be used in the experiment for anonymity. We let $\mathcal{O} = \{\mathsf{O}_{\mathsf{KG}}, \mathsf{O}_{\mathsf{UKG}}, \mathsf{O}_{\mathsf{GKG}}, \mathsf{O}^{\mathsf{A}}_{\mathsf{RSig}}, \mathsf{O}^{\mathsf{T}}_{\mathsf{RSig}}, \mathsf{O}_{\mathsf{Op}}\}$ and $\mathcal{O}^{\mathsf{T}} = \{\mathsf{O}_{\mathsf{KG}}, \mathsf{O}^{\mathsf{T}}_{\mathsf{GKG}}, \mathsf{O}^{\mathsf{A}}_{\mathsf{RSig}}, \mathsf{O}^{\mathsf{T}}_{\mathsf{RSig}}, \mathsf{O}_{\mathsf{Op}}\}$.

Note that we introduce O_{KG} to strengthen the security model. Specifically, $\mathsf{O}_{\mathsf{UKG}}$ and $\mathsf{O}_{\mathsf{GKG}}$ capture scenarios in which a malicious user or a malicious GM participates in the key generation process. However, there are situations in which a malicious user may obtain the public keys of other honest users.

Formal Descriptions. We define security requirements of PARS. Their experiments are listed in Fig. 1 and Fig. 2.

We first define non-frameability followed by traceability soundness since they are defined by the same experiment, apart from their winning conditions.

Definition 9. *A PARS scheme Π is non-frameable if for any sufficiently large security parameter λ and any PPT adversary $\mathcal{A}$, it holds that*

$\mathsf{ExpNFrm}_{\Pi,\mathcal{A}}(\lambda)$

$\mathsf{reg}, \mathsf{L_{KG}}, \mathsf{L_{HU}}, \mathsf{L_{CU}}, \mathsf{L_{SK}}, \mathsf{L^{A}_{Sig}}, \mathsf{L^{T}_{Sig}} \leftarrow \emptyset$;
$(\mathsf{pp}, \mathsf{msk}, \mathsf{oak}) \leftarrow \Pi.\mathsf{Set}(1^\lambda)$;
$(\mathsf{R}^* = (\mathsf{upk}_i^*)_{i\in[n]}, \mathsf{m}^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pp}, \mathsf{msk}, \mathsf{oak})$;
$\mathsf{id}^* = \Pi.\mathsf{Open}(\mathsf{pp}, \mathsf{oak}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*)$;
if $\mathsf{id}^* \neq 0 \wedge \mathsf{upk}_{\mathsf{id}^*} \in \mathsf{R}^* \wedge \mathsf{upk}_{\mathsf{id}^*} \in \mathsf{L_{KG}} \cup \mathsf{L_{HU}} \wedge \Pi.\mathsf{Vrf}(\mathsf{pp}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) = 1$
  $\wedge (\mathsf{upk}_{\mathsf{id}^*}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) \notin \mathsf{L^{T}_{Sig}} \wedge (\mathsf{upk}_{\mathsf{id}^*}, \mathsf{R}^*, \mathsf{m}^*, \cdot, \sigma^*) \notin \mathsf{L^{A}_{Sig}}$ **then return** 1
return 0

$\mathsf{ExpTrcSo}_{\Pi,\mathcal{A}}(\lambda)$

$\mathsf{reg}, \mathsf{L_{KG}}, \mathsf{L_{HU}}, \mathsf{L_{CU}}, \mathsf{L_{SK}}, \mathsf{L^{A}_{Sig}}, \mathsf{L^{T}_{Sig}} \leftarrow \emptyset$;
$(\mathsf{pp}, \mathsf{msk}, \mathsf{oak}) \leftarrow \Pi.\mathsf{Set}(1^\lambda)$;
$(\mathsf{R}^* = (\mathsf{upk}_i^*)_{i\in[n]}, \mathsf{m}^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pp}, \mathsf{msk}, \mathsf{oak})$;
$\mathsf{id}^* = \Pi.\mathsf{Open}(\mathsf{pp}, \mathsf{oak}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*)$;
if $\Pi.\mathsf{Vrf}(\mathsf{pp}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) = 0 \vee \mathsf{id}^* = 0 \vee (\mathsf{upk}_{\mathsf{id}^*}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) \in \mathsf{L^{T}_{Sig}}$
  $\vee (\mathsf{upk}_{\mathsf{id}^*}, \mathsf{R}^*, \mathsf{m}^*, \cdot, \sigma^*) \in \mathsf{L^{A}_{Sig}}$ **then return** 0
if $\forall i \in [n], \mathsf{id}_i^* \neq \mathsf{id}^* \wedge \mathsf{upk}_{\mathsf{id}^*} \notin \mathsf{L_{CU}}$ **then return** 1
return 0

$\mathsf{ExpEUFPARS}_{\Pi,\mathcal{A}}(\lambda)$:

$\mathsf{reg}, \mathsf{L_{KG}}, \mathsf{L_{CU}}, \mathsf{L_{SK}}, \mathsf{L^{A}_{Sig}}, \mathsf{L^{T}_{Sig}} \leftarrow \emptyset$;
$(\mathsf{pp}, \mathsf{msk}, \mathsf{oak}) \leftarrow \Pi.\mathsf{Set}(1^\lambda)$;
$(\mathsf{R}^* = (\mathsf{upk}_i^*)_{i\in[n]}, \mathsf{m}^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}\setminus\{\mathsf{O_{UKG}}\}}(\mathsf{pp})$;
if $\forall i \in [n], \mathsf{upk}_i^* \in \mathsf{L_{KG}} \wedge \Pi.\mathsf{Vrf}(\mathsf{pp}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) = 1 \wedge (\cdot, \mathsf{R}^*, \mathsf{m}^*, \cdot, \sigma^*) \notin \mathsf{L^{A}_{Sig}}$
  $\wedge (\cdot, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) \notin \mathsf{L^{T}_{Sig}}$ **then return** 1
return 0

Fig. 1. Non-frameability, traceability soundness, and unforgeability games for PARS.

$\Pr[\mathsf{ExpNFrm}_{\Pi,\mathcal{A}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$ *where* ExpNFrm *is the experiment defined in Fig. 1.*

Definition 10. *A PARS scheme Π satisfies traceability soundness if for any sufficiently large security parameter λ and any PPT adversary $\mathcal{A}$, it holds that* $\Pr[\mathsf{ExpTrcSo}_{\Pi,\mathcal{A}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$ *where* ExpTrcSo *is the experiment defined in Fig. 1.*

We next define unforgeability. Observe that the winning conditions of the definitions of non-frameability, traceability soundness, and traceability (that will be defined in Definition 14) do not completely capture the adversary considered in unforgeability.

Definition 11. *A PARS scheme Π is unforgeable if for any sufficiently large security parameter λ and any PPT adversary $\mathcal{A}$, it holds that* $\Pr[\mathsf{ExpEUFPARS}_{\Pi,\mathcal{A}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$ *where* ExpEUFPARS *is the experiment defined in Fig. 1.*

The next requirements are anonymity-related properties. Recall that we define two types of anonymities. Observe that, in ExpCAnon, the challenger is always asked to output an anonymous signature.

Definition 12. *A PARS scheme Π is honest-OA anonymous if for any sufficiently large security parameter λ and any PPT adversary $\mathcal{A}$, it holds that* $|\Pr[\mathsf{ExpHAnon}_{\Pi,\mathcal{A}}(\lambda) = 1] - 1/2| \leq \mathsf{negl}(\lambda)$ *where* ExpHAnon *is the experiment defined in Fig. 2.*

Definition 13. *A PARS scheme Π is corrupted-OA anonymous if for any sufficiently large security parameter λ and any PPT adversary $\mathcal{A}$, it holds that* $|\Pr[\mathsf{ExpCAnon}_{\Pi,\mathcal{A}}(\lambda) = 1] - 1/2| \leq \mathsf{negl}(\lambda)$ *where* ExpCAnon *is the experiment defined in Fig. 2.*

Finally, we discuss traceability. This requires that, for malicious signers who only know traceable keys, it is hard to create a signature that cannot be traced.

Definition 14. *A PARS scheme Π is traceable if for any sufficiently large security parameter λ and any PPT adversary $\mathcal{A}$, it holds that* $\Pr[\mathsf{ExpTrc}_{\Pi,\mathcal{A}}(\lambda) = 1] \leq \mathsf{negl}(\lambda)$ *where* ExpTrc *is the experiment defined in Fig. 2.*

Remark 1. At first attempt, we considered defining the "mode-hiding" property, which guarantees that it is hard for malicious signers and GM to distinguish if a signature is created by a signer with $f(\mathsf{id}) = 1$ or $f(\mathsf{id}) = 0$. However, we found that our definition of anonymity naturally captures this requirement.

4 Construction of PARS

In this section, we provide our construction of PARS.

4.1 Construction

In this section, we present the description of a generic construction Π of a PARS scheme from a PKE scheme PKE, a (standard) signature scheme SIG, a one-time signature scheme OTS, and a NIZKPoK ZK. As mentioned in Section 1.2, the language that ZK proves plays a central role in our construction. Thus, before providing the construction, we formally introduce a language $\mathcal{L}$ for ZK, where $\mathsf{R} = \{(\mathsf{id}_i, \mathsf{vk}_i)\}_{i\in[n]}$ is a ring:

$$\begin{aligned}
\mathcal{L} = \{&\mathsf{R}, \mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, c, \mathsf{ovk} \mid \exists \mathsf{upk}_{\mathsf{id}^*} = (\mathsf{id}^*, \mathsf{vk}_{\mathsf{id}^*}) \in \mathsf{R}, \mathsf{cert}_{\mathsf{id}^*}, \sigma_{\mathsf{SIG}}, \mathsf{r}_{\mathsf{PKE}} \text{ s.t.} \\
&\mathsf{SIG.Vrf}(\mathsf{vk}_{\mathsf{GM}}, \mathsf{id}^* || \mathsf{vk}_{\mathsf{id}^*} || f(\mathsf{id}^*), \mathsf{cert}_{\mathsf{id}^*}) = 1 \wedge \mathsf{SIG.Vrf}(\mathsf{vk}_{\mathsf{id}^*}, \mathsf{ovk}, \sigma_{\mathsf{SIG}}) = 1 \\
&\wedge ((f(\mathsf{id}^*) = 1 \wedge (c = \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{OA}}, \mathsf{id}^*; \mathsf{r}_{\mathsf{PKE}}) \vee c = \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{OA}}, 0; \mathsf{r}_{\mathsf{PKE}})))) \\
&\quad \vee (f(\mathsf{id}^*) = 0 \wedge c = \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{OA}}, \mathsf{id}^*; \mathsf{r}_{\mathsf{PKE}})))\}
\end{aligned}$$

$\mathsf{ExpHAnon}_{\Pi,\mathcal{A}}(\lambda)$

$\mathsf{reg}, \mathsf{L_{KG}}, \mathsf{L_{HU}}, \mathsf{L_{CU}}, \mathsf{L_{SK}}, \mathsf{L^A_{Sig}}, \mathsf{L^T_{Sig}} \leftarrow \emptyset$;
$(\mathsf{pp}, \mathsf{msk}, \mathsf{oak}) \leftarrow \Pi.\mathsf{Set}(1^\lambda)$;
$(\mathsf{R}^* = (\mathsf{upk}^*_i)_{i\in[n]}, \mathsf{usk}^*_{i_0}, \mathsf{usk}^*_{i_1}, \mathsf{m}^*) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pp}, \mathsf{msk})$;
if ($\mathsf{upk}^*_{i_0}$ and $\mathsf{upk}^*_{i_1}$ cannot be retrieved from $\mathsf{L_{SK}}$ and reg)
  $\vee\ \mathsf{upk}^*_{i_0} \notin \mathsf{R}^* \vee \mathsf{upk}^*_{i_1} \notin \mathsf{R}^*$ **then return** a random bit
$b \leftarrow \{0,1\}$;
if $f(\mathsf{id}^*_{i_b}) = 1$ **then**
  $b' \leftarrow \{0,1\}$;
  $\sigma^* \leftarrow \Pi.\mathsf{RSig}^{\mathsf{A}}(\mathsf{pp}, \mathsf{usk}^*_{i_b}, \mathsf{R}^*, \mathsf{m}^*, b')$;
else $\sigma^* \leftarrow \Pi.\mathsf{RSig}^{\mathsf{T}}(\mathsf{pp}, \mathsf{usk}^*_{i_b}, \mathsf{R}^*, \mathsf{m}^*)$
$b^\dagger \leftarrow \mathcal{A}^{\{\mathcal{O}\cup\mathsf{O}^{\sigma^*}_{\mathsf{Op}}\}\setminus\{\mathsf{O}_{\mathsf{Op}}\}}(\mathsf{pp}, \mathsf{msk}, \sigma^*)$
if $b^\dagger = b$ **then return** 1
return 0

$\mathsf{ExpCAnon}_{\Pi,\mathcal{A}}(\lambda)$

$\mathsf{reg}, \mathsf{L_{KG}}, \mathsf{L_{HU}}, \mathsf{L_{CU}}, \mathsf{L_{SK}}, \mathsf{L^A_{Sig}}, \mathsf{L^T_{Sig}} \leftarrow \emptyset$;
$(\mathsf{pp}, \mathsf{msk}, \mathsf{oak}) \leftarrow \Pi.\mathsf{Set}(1^\lambda)$;
$(\mathsf{R}^* = (\mathsf{upk}^*_i)_{i\in[n]}, \mathsf{usk}^*_{i_0}, \mathsf{usk}^*_{i_1}, \mathsf{m}^*) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pp}, \mathsf{msk}, \mathsf{oak})$;
if $f(\mathsf{id}^*_{i_0}) \neq 1 \vee f(\mathsf{id}^*_{i_1}) \neq 1 \vee \mathsf{upk}^*_{i_0} \notin \mathsf{R}^* \vee \mathsf{upk}^*_{i_1} \notin \mathsf{R}^*$
  $\vee$ ($\mathsf{upk}^*_{i_0}$ and $\mathsf{upk}^*_{i_1}$ cannot be retrieved from $\mathsf{L_{SK}}$ and reg) **then**
  return a random bit
$b \leftarrow \{0,1\}$;
$\sigma^* \leftarrow \Pi.\mathsf{RSig}^{\mathsf{A}}(\mathsf{pp}, \mathsf{usk}^*_{i_b}, \mathsf{R}^*, \mathsf{m}^*, 1)$;
$b^\dagger \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{pp}, \mathsf{msk}, \mathsf{oak}, \sigma^*)$
if $b^\dagger = b$ **then return** 1
return 0

$\mathsf{ExpTrc}_{\Pi,\mathcal{A}}(\lambda)$

$\mathsf{reg}, \mathsf{L_{KG}}, \mathsf{L_{HU}}, \mathsf{L_{CU}}, \mathsf{L_{SK}}, \mathsf{L^A_{Sig}}, \mathsf{L^T_{Sig}} \leftarrow \emptyset$;
$(\mathsf{pp}, \mathsf{msk}, \mathsf{oak}) \leftarrow \Pi.\mathsf{Set}(1^\lambda)$;
$(\mathsf{R}^* = (\mathsf{upk}^*_i)_{i\in[n]}, \mathsf{m}^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}^{\mathsf{T}}}(\mathsf{pp})$;
if $\Pi.\mathsf{Vrf}(\mathsf{pp}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) = 1 \wedge \forall \mathsf{upk_{id}} \in \mathsf{L_{KG}}$ s.t. $f(\mathsf{id}) = 1$
  $\wedge\ (\mathsf{upk_{id}}, \mathsf{R}^*, \mathsf{m}^*, 1, \sigma^*) \notin \mathsf{L^A_{Sig}} \wedge \Pi.\mathsf{Open}(\mathsf{pp}, \mathsf{oak}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) = 0$ **then**
  return 1
return 0

Fig. 2. Anonymity and traceability games for PARS.

In words, ZK for $\mathcal{L}$ proves that "it is generated by a signer in the ring R who has a valid certificate, and it encrypts the signer's identifier in a certified manner." We remark that the language $\mathcal{L}$ is crucial to our construction and is carefully defined to guarantee that the scheme constitutes a PARS. We will elaborate on this point in the description of the signing algorithm. Namely, the first condition on the standard signature is necessary to prove the security.

The generic construction Π of a PARS scheme is provided in Fig. 3 and Fig. 4, where Fig. 4 describes the interactive key generation algorithm and Fig. 3 the remaining algorithms. We briefly review the construction as follows.

The setup algorithm $\Pi.\mathsf{Set}$ creates a master secret key msk and the opening authority key oak along with a public parameter pp. The master secret key is a signing key of SIG, which is provided to the GM, and oak is a secret key of PKE, which is provided to the OA.

In the key generation algorithm $\Pi.\langle \mathsf{GKG}(\mathsf{msk}, \mathsf{reg}), \mathsf{UKG}(\mathsf{id})\rangle(\mathsf{pp})$, the user first sends his identifier id and verification key vk of SIG that is obtained locally. Then the GM publishes a certificate based on the value $f(\mathsf{id})$ if id and vk are fresh ones. We assume that the interaction is done via a secure channel. In some group signature constructions, a proof of possession is required during the joining protocol to prevent rogue-key registration. In our construction, however, possession of the corresponding signing key is already enforced at signing time by the NIZKPoK relation, which requires a valid signature under the registered verification key. Therefore, this additional proof of possession is not needed for our security proof, and we omit it for simplicity.

There are two signing algorithms, but they operate in almost the same manner. We emphasize that the proof generated by ZK plays the essential role as the signature. That is, signers are free to choose a ring because ZK for $\mathcal{L}$ allows such a selection. Furthermore, we can switch the anonymous/traceable mode due to the conditions of $\mathcal{L}$. Note that the ciphertext is required for traceability, and we need to use OTS to avoid the attack that malicious signers re-use a valid proof.

We do not explain $\Pi.\mathsf{Vrf}$ and $\Pi.\mathsf{Open}$ in detail. An important point is that the opening algorithm decrypts the given ciphertext only when the given signature is valid.

4.2 Security Analysis

In this section, we demonstrate that the construction Π for a binary function family $\mathcal{F}$ satisfies security requirements. Before the proof, we describe how key generation oracles work for Π, as they were defined in an abstract manner in the oracle description. Let $\mathsf{pp} := (\mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{crs})$, $\mathsf{msk} := \mathsf{sigk}_{\mathsf{GM}}$, and $\mathsf{oak} := \mathsf{sk}_{\mathsf{OA}}$ be values created by $\Pi.\mathsf{Set}(\mathcal{F})$. Further, the lists $\mathsf{reg}, \mathsf{L}_{\mathsf{KG}}, \mathsf{L}_{\mathsf{HU}}, \mathsf{L}_{\mathsf{CU}}, \mathsf{L}_{\mathsf{SK}}, \mathsf{L}^{\mathsf{A}}_{\mathsf{Sig}}$, and $\mathsf{L}^{\mathsf{T}}_{\mathsf{Sig}}$ are initially empty sets. The description of key generation oracles is below.

$\mathsf{O}_{\mathsf{KG}}(\mathsf{id})$**:** Given an identifier id, if $(\mathsf{id}, \cdot) \in \mathsf{L}_{\mathsf{KG}} \cup \mathsf{L}_{\mathsf{HU}} \cup \mathsf{L}_{\mathsf{CU}}$, then return $\perp$. It first creates $(\mathsf{sigk}, \mathsf{vk}) \leftarrow \mathsf{SIG.KG}(1^\lambda)$ where vk is not in reg. Then, it creates $\mathsf{cert} \leftarrow \mathsf{SIG.Sig}(\mathsf{sigk}_{\mathsf{GM}}, \mathsf{id}||\mathsf{vk}||f(\mathsf{id}))$, and checks if $\mathsf{SIG.Vrf}(\mathsf{vk}_{\mathsf{GM}}, \mathsf{id}||\mathsf{vk}||f(\mathsf{id}), \mathsf{cert}) = 1$. If the verification fails, then it repeats the same procedure until it

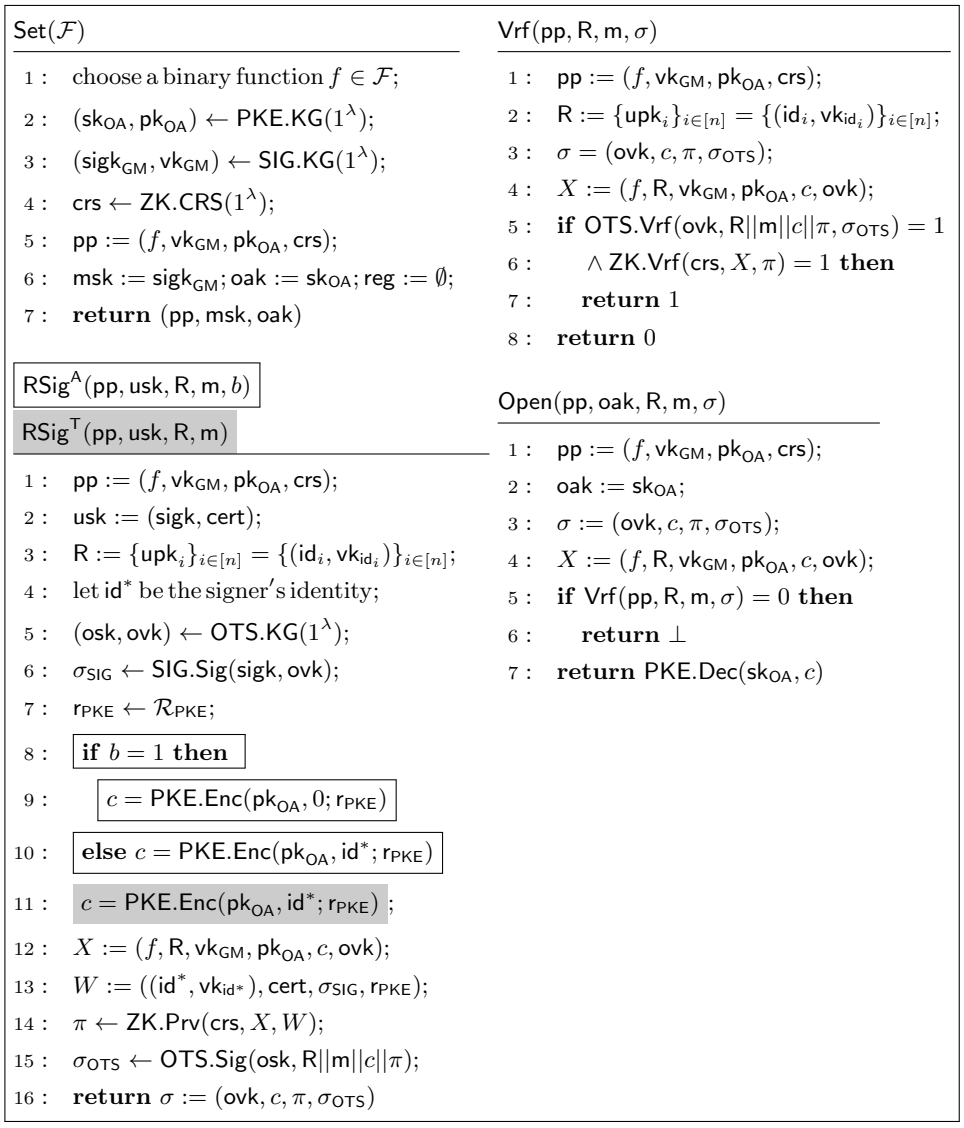

Fig. 3. The construction Π of our PARS scheme, except for the key generation algorithm.

creates a valid signature. Otherwise, it sets $\mathsf{trans} := (\mathsf{id}, \mathsf{vk}, \mathsf{cert})$, $\mathsf{reg} := \mathsf{reg} \cup \{\mathsf{trans}\}$, $\mathsf{usk} := (\mathsf{sigk}, \mathsf{cert})$, $\mathsf{upk} := (\mathsf{id}, \mathsf{vk})$, $\mathsf{L_{KG}} := \mathsf{L_{KG}} \cup \{\mathsf{upk}\}$, and $\mathsf{L_{SK}} := \mathsf{L_{SK}} \cup \{\mathsf{id}, \mathsf{usk}\}$, and outputs upk.

$\mathsf{O_{UKG}}(\mathsf{id})$**:** Given an identifier id, if $(\mathsf{id}, \cdot) \in \mathsf{L_{KG}} \cup \mathsf{L_{HU}} \cup \mathsf{L_{CU}}$, then return $\perp$. Otherwise, it creates $(\mathsf{sigk}, \mathsf{vk}) \leftarrow \mathsf{SIG.KG}(1^\lambda)$ and sends $(\mathsf{id}, \mathsf{vk})$ to the adversary. The adversary returns $\mathsf{trans} := (\mathsf{id}, \mathsf{vk}, \mathsf{cert})$. Finally, the oracle sets

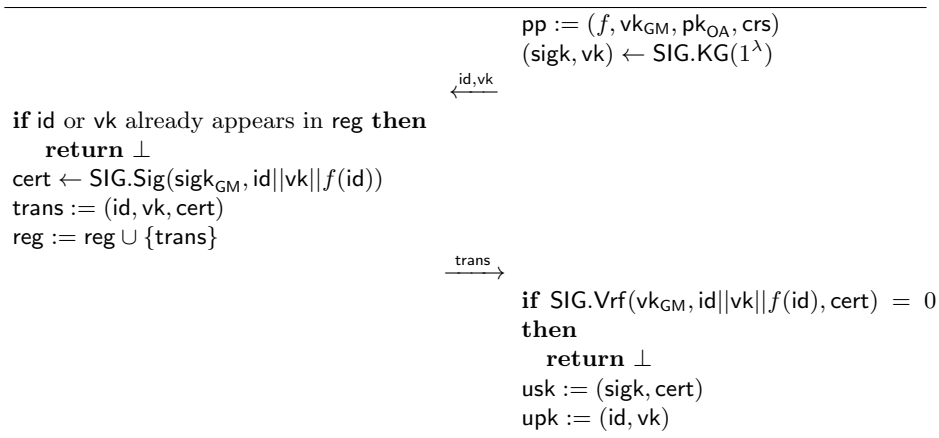

Fig. 4. The interactive key generation algorithm GKG and UKG of the construction Π.

$\mathsf{reg} := \mathsf{reg} \cup \{\mathsf{trans}\}, \mathsf{usk} := (\mathsf{sigk}, \mathsf{cert}), \mathsf{upk} := (\mathsf{id}, \mathsf{vk}), \mathsf{L}_{\mathsf{HU}} := \mathsf{L}_{\mathsf{HU}} \cup \{\mathsf{upk}\}$, and $\mathsf{L}_{\mathsf{SK}} := \mathsf{L}_{\mathsf{SK}} \cup \{\mathsf{id}, \mathsf{usk}\}$ if $\mathsf{SIG.Vrf}(\mathsf{vk}_{\mathsf{GM}}, \mathsf{id}||\mathsf{vk}||f(\mathsf{id}), \mathsf{cert}) = 1$. Otherwise, return $\perp$.

$\mathsf{O}_{\mathsf{GKG}}$**:** The adversary first sends $(\mathsf{id}, \mathsf{vk})$ to the oracle. If $(\mathsf{id}, \cdot) \in \mathsf{L}_{\mathsf{KG}} \cup \mathsf{L}_{\mathsf{HU}} \cup \mathsf{L}_{\mathsf{CU}}$, then the oracle returns $\perp$. Otherwise, the oracle creates $\mathsf{cert} \leftarrow \mathsf{SIG.Sig}(\mathsf{sigk}_{\mathsf{GM}}, \mathsf{id}||\mathsf{vk}||f(\mathsf{id}))$, and checks if $\mathsf{SIG.Vrf}(\mathsf{vk}_{\mathsf{GM}}, \mathsf{id}||\mathsf{vk}||f(\mathsf{id}), \mathsf{cert}) = 1$. If the verification fails, then it repeats the same procedure until it creates a valid signature. Otherwise, it sets $\mathsf{trans} := (\mathsf{id}, \mathsf{vk}, \mathsf{cert}), \mathsf{reg} := \mathsf{reg} \cup \{\mathsf{trans}\}$ and $\mathsf{L}_{\mathsf{CU}} := \mathsf{L}_{\mathsf{CU}} \cup \{\mathsf{upk} = (\mathsf{id}, \mathsf{vk})\}$, and outputs trans.

Now, we prove the security.

Lemma 1. *The construction Π is non-frameable if* OTS *is strongly unforgeable,* SIG *is unforgeable, and* ZK *is zero-knowledge and simulation-sound extractable.*

Proof. We provide an overview of the proof before the formal description. We first describe the experiment $\mathsf{ExpNFrm}_{\Pi,\mathcal{A}}$ between a PPT adversary $\mathcal{A}$ as Hybrid 0. Next, we introduce Hybrid 1 such that the challenger runs $\mathsf{ZK.ExtCRS}$ instead of $\mathsf{ZK.CRS}$ at the setup phase, and runs $\mathsf{ZK.Ext}$ at the end of the game. Then, we introduce events regarding the result of the game and the extraction result. Finally, we show that, if some events happen with non-negligible probability, then we can construct a PPT adversary $\mathcal{B}$ that breaks the security of building blocks. Namely, $\mathcal{B}$ simulates $\mathsf{ExpNFrm}_{\Pi,\mathcal{A}}$ as the challenger while running $\mathcal{A}$ as an adversary. We begin with the description of Hybrid 0.

Hybrid 0. The non-frameability game between the challenger and $\mathcal{A}$. In the setup phase, the challenger sets up $\mathsf{ExpNFrm}$ as follows. First, it chooses a function $f \in \mathcal{F}$, initializes the lists $\mathsf{reg}, \mathsf{L}_{\mathsf{KG}}, \mathsf{L}_{\mathsf{HU}}, \mathsf{L}_{\mathsf{CU}}, \mathsf{L}_{\mathsf{SK}}, \mathsf{L}^{\mathsf{A}}_{\mathsf{Sig}}$, and $\mathsf{L}^{\mathsf{T}}_{\mathsf{Sig}}$ as empty sets.

Next, it computes $(\mathsf{sk}_{\mathsf{OA}}, \mathsf{pk}_{\mathsf{OA}}) \leftarrow \mathsf{PKE.KG}(1^\lambda), (\mathsf{sigk}_{\mathsf{GM}}, \mathsf{vk}_{\mathsf{GM}}) \leftarrow \mathsf{SIG.KG}(1^\lambda)$, and $\mathsf{crs} \leftarrow \mathsf{ZK.CRS}(1^\lambda)$, sets $\mathsf{pp} := (f, \mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{crs}), \mathsf{msk} := \mathsf{sigk}_{\mathsf{GM}}$, and $\mathsf{oak} := \mathsf{sk}_{\mathsf{OA}}$, and provides $(\mathsf{pp}, \mathsf{msk}, \mathsf{oak})$ to the adversary $\mathcal{A}$.

When $\mathcal{A}$ makes queries to oracles in the challenge phase, the challenger returns the answer as follows.

$\mathsf{O}_{\mathsf{KG}}, \mathsf{O}_{\mathsf{UKG}}$, and $\mathsf{O}_{\mathsf{GKG}}$: Follow the description of these oracles.

$\mathsf{O}^{\mathsf{A}}_{\mathsf{RSig}}$: Given a public key $\mathsf{upk} = (\mathsf{id}, \mathsf{vk})$, a ring $\mathsf{R} = (\mathsf{id}_i, \mathsf{vk}_{\mathsf{id}_i})_{i\in[n]}$, a message m, and a bit $b \in \{0,1\}$, the challenger outputs $\perp$ if there is no entry s.t. $(\mathsf{id}, \mathsf{usk}) \in \mathsf{L}_{\mathsf{SK}}$ where $\mathsf{usk} = (\mathsf{sigk}, \mathsf{cert})$, or $\mathsf{upk} \notin \mathsf{R}$. Otherwise, it computes $(\mathsf{osk}, \mathsf{ovk}) \leftarrow \mathsf{OTS.KG}(1^\lambda)$ and $\sigma_{\mathsf{SIG}} \leftarrow \mathsf{SIG.Sig}(\mathsf{sigk}, \mathsf{ovk})$. Next, it chooses $\mathsf{r}_{\mathsf{PKE}} \leftarrow \mathcal{R}_{\mathsf{PKE}}$, and computes $c = \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{OA}}, \mathsf{id}; \mathsf{r}_{\mathsf{PKE}})$ (if $b = 0$) or $c = \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{OA}}, 0; \mathsf{r}_{\mathsf{PKE}})$ (if $b = 1$). For $X = (f, \mathsf{R}, \mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, c, \mathsf{ovk})$ and $W = ((\mathsf{id}, \mathsf{vk}), \mathsf{cert}, \sigma_{\mathsf{SIG}}, \mathsf{r}_{\mathsf{PKE}})$, compute $\pi \leftarrow \mathsf{ZK.Prv}(\mathsf{crs}, X, W)$. Finally, the challenger computes $\sigma_{\mathsf{OTS}} \leftarrow \mathsf{OTS.Sig}(\mathsf{osk}, \mathsf{R}||\mathsf{m}||c||\pi)$, outputs $\sigma := (\mathsf{ovk}, c, \pi, \sigma_{\mathsf{OTS}})$, and updates $\mathsf{L}^{\mathsf{A}}_{\mathsf{Sig}} := \mathsf{L}^{\mathsf{A}}_{\mathsf{Sig}} \cup \{(\mathsf{upk}, \mathsf{R}, \mathsf{m}, b, \sigma)\}$.

$\mathsf{O}^{\mathsf{T}}_{\mathsf{RSig}}$: Given a public key $\mathsf{upk} = (\mathsf{id}, \mathsf{vk})$, a ring $\mathsf{R} = (\mathsf{id}_i, \mathsf{vk}_{\mathsf{id}_i})_{i\in[n]}$, and a message m, it behaves as in $\mathsf{O}^{\mathsf{A}}_{\mathsf{RSig}}$ except that it always computes $c = \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{OA}}, \mathsf{id}; \mathsf{r}_{\mathsf{PKE}})$. Finally, the challenger updates $\mathsf{L}^{\mathsf{T}}_{\mathsf{Sig}} := \mathsf{L}^{\mathsf{T}}_{\mathsf{Sig}} \cup \{(\mathsf{upk}, \mathsf{R}, \mathsf{m}, \sigma)\}$.

O_{Op}: Given a ring $\mathsf{R} = (\mathsf{id}_i, \mathsf{vk}_{\mathsf{id}_i})_{i\in[n]}$, a message m, and a signature $\sigma = (\mathsf{ovk}, c, \pi, \sigma_{\mathsf{OTS}})$, it outputs $\mathsf{PKE.Dec}(\mathsf{sk}_{\mathsf{OA}}, c)$ if $\mathsf{OTS.Vrf}(\mathsf{ovk}, \mathsf{R}||\mathsf{m}||c||\pi, \sigma_{\mathsf{OTS}}) = 1$, and $\mathsf{ZK.Vrf}(\mathsf{crs}, X, \pi) = 1$ where $X = (f, \mathsf{R}, \mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, c, \mathsf{ovk})$. Otherwise 0.

When $\mathcal{A}$ outputs a ring $\mathsf{R}^* = (\mathsf{id}^*_i, \mathsf{vk}^*_{\mathsf{id}_i},)_{i\in[n]}$, a message m^*, and a signature $\sigma^* = (\mathsf{ovk}^*, c^*, \pi^*, \sigma^*_{\mathsf{OTS}})$, the challenger first checks if $\mathsf{OTS.Vrf}(\mathsf{ovk}^*, \mathsf{R}^*||\mathsf{m}^*||c^*||\pi^*, \sigma^*_{\mathsf{OTS}}) = 1$ and $\mathsf{ZK.Vrf}(\mathsf{crs}, X^*, \pi^*) = 1$ where $X^* = (f, \mathsf{R}^*, \mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, c^*, \mathsf{ovk}^*)$ (i.e., the challenger checks the validity of σ^*). If not, it outputs 0. Otherwise, it computes $\mathsf{PKE.Dec}(\mathsf{sk}_{\mathsf{OA}}, c^*) = \mathsf{id}^*$, and outputs 0 if $\mathsf{id}^* = 0$. Finally, if there is $i \in [n]$ s.t. $\mathsf{id}^*_i = \mathsf{id}^*, \mathsf{upk}_{\mathsf{id}^*} \in \mathsf{L}_{\mathsf{KG}} \cup \mathsf{L}_{\mathsf{HU}}$, $(\mathsf{upk}_{\mathsf{id}^*}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) \notin \mathsf{L}^{\mathsf{T}}_{\mathsf{Sig}}$ and $(\mathsf{upk}_{\mathsf{id}^*}, \mathsf{R}^*, \mathsf{m}^*, \cdot, \sigma^*) \notin \mathsf{L}^{\mathsf{A}}_{\mathsf{Sig}}$, then output 1, otherwise 0.

Hybrid 1. Modify Hybrid 0 so that the challenger runs $(\mathsf{crs}, \tau, \zeta) \leftarrow \mathsf{ZK.ExtCRS}(1^\lambda)$ instead of $\mathsf{ZK.CRS}$ in the setup phase. Further, we let the challenger run $\mathsf{ZK.Ext}(\mathsf{crs}, \zeta, X^*, \pi^*)$ just before outputting the final bit.

Observe that the execution of $\mathsf{ZK.Ext}$ by the challenger does not affect the output of Hybrid 1. Let Succ_i be an event that the challenger outputs 1 in Hybrid i for $i \in \{0,1\}$, and Bad be an event that the challenger extracts a witness $W^* = ((\mathsf{id}^*, \mathsf{vk}^*), \mathsf{cert}^*, \sigma^*_{\mathsf{SIG}}, \mathsf{r}^*_{\mathsf{PKE}})$ such that $(X^*, W^*) \notin R_{\mathcal{L}}$ at the end of Hybrid 1. Observe that the advantage in $\mathsf{ExpNFrm}_{\Pi,\mathcal{A}}$ is equal to $\Pr[\mathsf{Succ}_0]$. Further, by the triangle inequality, it holds that $\Pr[\mathsf{Succ}_0] \leq |\Pr[\mathsf{Succ}_0] - \Pr[\mathsf{Succ}_1]| + \Pr[\mathsf{Succ}_1]$. It is obvious that $|\Pr[\mathsf{Succ}_0] - \Pr[\mathsf{Succ}_1]|$

is negligible due to the CRS indistinguishability of ZK. Thus, if we prove that $|\Pr[\mathsf{Succ}_1]|$ is negligible in λ, the proof is done.

To show the inequality above, we inspect $|\Pr[\mathsf{Succ}_1]|$ in detail, and we have $\Pr[\mathsf{Succ}_1] = \Pr[\mathsf{Succ}_1 \wedge \mathsf{Bad}] + \Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$. Thus, we evaluate $\Pr[\mathsf{Succ}_1 \wedge \mathsf{Bad}]$ and $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$, respectively.

Claim. If $\Pr[\mathsf{Succ}_1 \wedge \mathsf{Bad}]$ is non-negligible, then ZK is not simulation-sound extractable.

Proof. We construct a PPT adversary $\mathcal{B}$ that breaks the simulation-sound extractability of ZK by simulating $\mathcal{A}$.

Setup Phase. Upon receiving (crs, ζ) from the challenger, $\mathcal{B}$ sets up $\mathsf{ExpNFrm}$ in the same way as Hybrid 1, except that it uses the crs instead of generating it.

Challenge Phase. If $\mathcal{A}$ makes queries to oracles, $\mathcal{B}$ simulates the answer in the same way as Hybrid 1. When $\mathcal{A}$ outputs $\mathsf{R}^*, \mathsf{m}^*$, and σ^*, $\mathcal{B}$ outputs $X^* = (f, \mathsf{R}^*, \mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, c^*, \mathsf{ovk}^*)$, and a proof π^* that is contained in σ^*.

Analysis. Recall that Bad is the event that, for $W^* \leftarrow \mathsf{ZK.Ext}(\mathsf{crs}, \zeta, X^*, \pi^*)$, it holds that $(X^*, W^*) \notin R_{\mathcal{L}}$, and Succ_1 be an event that the challenger output 1 in Hybrid 1, i.e., it holds that $\mathsf{ZK.Vrf}(\mathsf{crs}, X^*, \pi^*) = 1$. Moreover, the simulation oracle in the simulation-sound extractability game is not used in the simulation of the hybrid. Therefore, we conclude that if $\mathsf{Bad} \wedge \mathsf{Succ}_1$ happens, then it indicates that the simulation-sound extractability of ZK is violated. □

What remains is to evaluate $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$. If $\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}$ happens, the challenger in Hybrid 1 can extract the correct witness $W^* = ((\mathsf{id}^*, \mathsf{vk}^*), \mathsf{cert}^*, \sigma^*_{\mathsf{SIG}}, \mathsf{r}^*_{\mathsf{PKE}})$. Note that $\mathcal{A}$ should obtain $\mathsf{upk}^* = (\mathsf{id}^*, \mathsf{vk}^*)$ via an oracle query due to the winning condition of $\mathsf{ExpNFrm}$, but does not obtain $\mathsf{usk}^* = (\mathsf{sigk}^*, \mathsf{cert}^*)$ from the oracle, because $\mathsf{upk}^* \in \mathsf{L}_{\mathsf{KG}} \cup \mathsf{L}_{\mathsf{HU}}$.

We now focus on σ^*_{SIG} $(= \mathsf{SIG.Sig}(\mathsf{sigk}^*, \mathsf{ovk}^*))$. This value is not contained in σ^*, but necessary to create the NIZKPoK proof π^*. If $\mathcal{A}$ is able to create it by itself (without receiving sigk^* from oracles), then we can construct a PPT adversary $\mathcal{B}$ that breaks unforgeability of SIG. That is, $\mathcal{B}$ simulates Hybrid 1 between $\mathcal{A}$ in the unforgeability game of SIG, and when $\mathcal{A}$ outputs σ^*, $\mathcal{B}$ extracts σ^*_{SIG} from it, and outputs σ^*_{SIG} as a forgery.

The above reduction works well if $\mathcal{B}$ does not query ovk^* (as a message) to the signing oracle in $\mathsf{ExpEUFSig}_{\mathsf{Sig},\mathcal{B}}$ (this will be shown in the paragraph "Reduction to Unforgeability of SIG"). The problem is that we cannot avoid the possibility that $\mathcal{B}$ makes such a query, and thus it is unclear if the reduction above works. However, in this case, we argue that $\mathcal{A}$ somehow forges a one-time signature with respect to ovk^*, and thus strong unforgeability of OTS is violated (this will be shown in the paragraph "Reduction to Strong Unforgeability of OTS"). In what follows, we formally prove these two cases.

Reduction to Unforgeability of SIG. Recall that we are under the assumption that $\mathcal{B}$ does not query ovk^* (as a message) to the signing oracle. In this case, we prove that if $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$ is non-negligible, then we can construct a PPT adversary $\mathcal{B}$ that breaks unforgeability of SIG. The adversary $\mathcal{B}$ plays $\mathsf{ExpEUFSig}_{\mathsf{Sig},\mathcal{B}}$ as follows.

Setup Phase. Given a verification key vk from the challenger, $\mathcal{B}$ sets up $\mathsf{ExpNFrm}$ in the same way as Hybrid 1. This vk will be used later.

Forgery Phase. When $\mathcal{A}$ makes queries, $\mathcal{B}$ answers in the same way as Hybrid 1 except for the following exceptions. Without loss of generality, we assume that $\mathcal{A}$ obtains $q = \mathsf{poly}(\lambda)$ honest users' public keys via O_{KG} and $\mathsf{O}_{\mathsf{UKG}}$. We let $\mathcal{B}$ uniformly choose an index $i^* \in [q]$ in the setup phase, and when $\mathcal{A}$ makes the i^*-th query to O_{KG} or $\mathsf{O}_{\mathsf{UKG}}$ for the identifier id_{i^*}, $\mathcal{B}$ calls the signing oracle of the unforgeability game of SIG to obtain $\mathsf{cert}_{\mathsf{id}_{i^*}}$, and embeds the verification key vk in the public key. Moreover, if $\mathcal{A}$ makes signing queries that necessitate the signing key of id_{i^*}, $\mathcal{B}$ obtains the signature σ_{SIG} via the signing oracle (the remaining part of the signing procedure is the same as Hybrid 1).

When $\mathcal{A}$ outputs $\sigma^* = (\mathsf{ovk}^*, c^*, \pi^*, \sigma^*_{\mathsf{OTS}})$, $\mathcal{B}$ runs $\mathsf{ZK.Ext}$ and obtains σ^*_{SIG}. Finally, $\mathcal{B}$ outputs σ^*_{SIG} with ovk^*.

Analysis. As we are under the assumption that $\mathcal{B}$ does not obtain σ^*_{SIG} via the signing oracle of the underlying unforgeability game, $\mathcal{B}$ succeeds in breaking unforgeability of SIG if $\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}$ occurs and $\mathsf{id}^* = \mathsf{id}_{i^*}$. Thus, the advantage of $\mathcal{B}$ is at least $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]/q$, which is non-negligible if $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$ is non-negligible.

Reduction to Strong Unforgeability of OTS. We prove that if $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$ is non-negligible, then we can construct a PPT adversary $\mathcal{B}$ that breaks strong unforgeability of OTS in this case. The idea is similar to the proof of the first case; $\mathcal{B}$ uses the challenge verification key from the strong unforgeability game of OTS during the simulation of the signing oracle of $\mathsf{ExpNFrm}$ at a randomly chosen time. We assume without loss of generality that the challenge verification key will not duplicate during the simulation. We demonstrate the reduction as follows.

Setup Phase. Given a verification key ovk from the challenger, $\mathcal{B}$ sets up $\mathsf{ExpNFrm}$ in the same way as Hybrid 1.

Forgery Phase. When $\mathcal{A}$ makes queries, $\mathcal{B}$ answers in the same way as Hybrid 1 except for the following exceptions. Without loss of generality, we assume that $\mathcal{A}$ obtains $q = \mathsf{poly}(\lambda)$ signatures via the signing oracles, and $\mathcal{B}$ uniformly chooses an index $i^* \in [q]$ in the setup phase. Suppose that $\mathcal{A}$ makes i^*-th query for randomly chosen $i^* \in [q]$ to obtain a signature $\sigma^\dagger = (\mathsf{ovk}^\dagger, c^\dagger, \pi^\dagger, \sigma^\dagger_{\mathsf{OTS}})$ for a message $\mathsf{m}^\dagger$ and a ring $\mathsf{R}^\dagger$. When creating $\sigma^\dagger_{\mathsf{OTS}}$, $\mathcal{B}$ calls the signing oracle in the strong unforgeability game of OTS (that can be used only once) on $\mathsf{R}^\dagger||\mathsf{m}^\dagger||c^\dagger||\pi^\dagger$.

When $\mathcal{A}$ outputs $\sigma^* = (\mathsf{ovk}^*, c^*, \pi^*, \sigma^*_{\mathsf{OTS}})$, $\mathcal{B}$ runs $\mathsf{ZK.Ext}$ and obtains σ^*_{OTS} on the message $\mathsf{R}^*||\mathsf{m}^*||c^*||\pi^*$. Finally, $\mathcal{B}$ outputs σ^*_{OTS} with ovk^*.

Analysis. As we are under the assumption that $\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}$ occurs and $\mathcal{A}$ obtains ovk^* via an oracle query, if $\mathsf{ovk}^* = \mathsf{ovk}^\dagger = \mathsf{ovk}$, then $\mathcal{B}$ wins the unforgeability game. That is, since i^* is chosen uniformly, $\mathcal{B}$ wins the game with probability at least $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]/q$, which is non-negligible if $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$ is non-negligible.

To sum up the discussion, we conclude that $\mathit{\Pi}$ is non-frameable if OTS is strongly unforgeable, SIG is unforgeable, and ZK is zero-knowledge and simulation-sound extractable. □

Lemma 2. *The construction $\mathit{\Pi}$ satisfies traceability soundness if ZK is simulation-sound extractable.*

Proof. We assume for contradiction that there exists a PPT adversary $\mathcal{A}$ that violates traceability soundness of $\mathit{\Pi}$ with non-negligible probability. Then, we demonstrate the existence of a PPT reduction algorithm $\mathcal{R}$ that breaks the simulation-sound extractability of ZK, by simulating $\mathsf{ExpTrcSo}_{\mathit{\Pi},\mathcal{A}}$ as a challenger. The idea is that if $\mathcal{A}$ wins the game, the opening result should be invalid, i.e., the ciphertext that is contained in the output signature decrypts to an identifier outside of the ring. Thus, if $\mathcal{R}$ outputs the proof along with the statement, the extraction algorithm must fail to extract a correct witness, which violates the simulation-sound extractability.

The algorithm $\mathcal{R}$ behaves similarly to the challenger (say $\mathcal{CH}$) in Hybrid 0 of the proof of Lemma 1. (Recall that $\mathsf{ExpNFrm}$ and $\mathsf{ExpTrcSo}$ are defined in almost the same manner.) The only difference is that $\mathcal{R}$ uses the CRS that is provided by the challenger rather than creating it by itself. The description of $\mathcal{R}$ is as follows.

Setup Phase. The challenger first runs $(\mathsf{crs}, \tau, \zeta) \leftarrow \mathsf{ZK.ExtCRS}$ and provides crs and ζ to $\mathcal{R}$. Given crs and ζ, $\mathcal{R}$ sets up $\mathsf{ExpTrcSo}$ in the same manner as $\mathcal{CH}$, except that it uses the given crs instead of creating it.

Challenge Phase. When $\mathcal{A}$ makes queries to the oracles, $\mathcal{R}$ simulates the oracle answers in the same manner as $\mathcal{CH}$. When $\mathcal{A}$ outputs $\mathsf{R}^* = \{\mathsf{upk}_i\}_{i\in[n]} = \{(\mathsf{id}^*_i, \mathsf{vk}^*_i)\}_{i\in[n]}, \mathsf{m}^*$ and $\sigma^* = (\mathsf{ovk}^*, c^*, \pi^*, \sigma^*_{\mathsf{OTS}})$, $\mathcal{R}$ checks if $\mathit{\Pi}.\mathsf{Vrf}(\mathsf{pp}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) = 1$ and $\mathsf{PKE.Dec}(\mathsf{sk}_{\mathsf{OA}}, c^*) \neq 0$. If not, $\mathcal{R}$ outputs $\perp$, otherwise, $\mathcal{R}$ outputs the proof π^* along with the statement $X^* := (f, \mathsf{R}^*, \mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, c^*, \mathsf{ovk}^*)$. On receiving X^* and π^*, the challenger runs $\mathsf{ZK.Ext}(\mathsf{crs}, \zeta, X^*, \pi^*)$ to see if $X^* \in \mathcal{L}$.

Analysis. We argue that if $\mathcal{A}$ wins the game, $\mathcal{R}$ breaks the simulation-sound extractability of ZK. Let $\mathit{\Pi}.\mathsf{Open}(\mathsf{pp}, \mathsf{oak}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) = \mathsf{PKE.Dec}(\mathsf{sk}_{\mathsf{OA}}, c^*) = \mathsf{id}^*$. Due to the winning condition of $\mathsf{ExpTrcSo}$, it holds that $\forall i \in [n], \mathsf{id}^*_i \neq \mathsf{id}^*$ and $\mathsf{ZK.Vrf}(\mathsf{crs}, X^*, \pi^*) = 1$. Thus, X^* is a false statement but π^* is a valid proof

for X^*. Furthermore, observe that the simulation oracle in the simulation-sound extractability game is not used in the above simulation.

To sum up the above discussion, we conclude that if there exists a PPT adversary $\mathcal{A}$ that breaks traceability soundness of Π with non-negligible probability, then there exists a PPT reduction algorithm $\mathcal{R}$ that breaks the simulation-sound extractability of ZK with non-negligible probability. □

Lemma 3. *The construction Π is unforgeable if* OTS *is strongly unforgeable,* SIG *is unforgeable, and* ZK *is zero-knowledge and simulation-sound extractable.*

Proof. The proof follows the same approach as that of Lemma 1. For similar parts, we simplify the proof and focus only on the essential points.

Hybrid 0. Let Hybrid 0 be $\mathsf{ExpEUFPARS}_{\Pi,\mathcal{A}}$ for a PPT adversary $\mathcal{A}$. The oracles are simulated as in the proof of Lemma 1, but $\mathsf{O}_{\mathsf{UKG}}$ is not used since msk and oak are not provided to $\mathcal{A}$.

Hybrid 1. (The same as Hybrid 1 in the proof of Lemma 1.) Modify Hybrid 0 so that the challenger runs $(\mathsf{crs}, \tau, \zeta) \leftarrow \mathsf{ZK.ExtCRS}(1^\lambda)$ instead of $\mathsf{ZK.CRS}$ in the setup phase. Further, we let the challenger run $\mathsf{ZK.Ext}(\mathsf{crs}, \zeta, X^*, \pi^*)$ just before outputting the final bit.

Let P_i be the probability that the challenger outputs 1 in Hybrid i for $i \in \{0,1\}$, and it follows that $|P_1 - P_0|$ is negligible due to the CRS indistinguishability of ZK. Let Succ_i be an event that the challenger outputs 1 in Hybrid i for $i \in \{0,1\}$, and Bad be an event that the challenger obtains a witness $W^* = ((\mathsf{id}^*, \mathsf{vk}^*), \mathsf{cert}^*, \sigma^*_{\mathsf{SIG}}, \mathsf{r}^*_{\mathsf{PKE}})$ such that $(X^*, W^*) \notin R_{\mathcal{L}}$, at the end of Hybrid 1. Observe that the advantage in $\mathsf{ExpEUFPARS}_{\Pi,\mathcal{A}}$ is equal to $\Pr[\mathsf{Succ}_0]$. As $\Pr[\mathsf{Succ}_0] \leq |\Pr[\mathsf{Succ}_0] - \Pr[\mathsf{Succ}_1]| + \Pr[\mathsf{Succ}_1]$, and $|\Pr[\mathsf{Succ}_0] - \Pr[\mathsf{Succ}_1]|$ is negligible, the lemma follows if $\Pr[\mathsf{Succ}_1]$ is negligible. By a similar discussion in the proof of Lemma 1, we have $\Pr[\mathsf{Succ}_1] \leq \Pr[\mathsf{Succ}_1 \wedge \mathsf{Bad}] + \Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$, and $\Pr[\mathsf{Succ}_1 \wedge \mathsf{Bad}]$ is negligible if ZK is simulation-sound extractable.

Now, we evaluate $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$. Let $(\mathsf{R}^* = \{(\mathsf{id}^*_i, \mathsf{vk}^*_i)\}_{i\in[n]}, \mathsf{m}^*, \sigma^*)$ be the output of $\mathcal{A}$. Although $\mathsf{ExpEUFPARS}$ does not care about the result of $\Pi.\mathsf{Open}(\mathsf{pp}, \mathsf{oak}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) = \mathsf{id}^*$, we distinguish cases based on id^*. As no id^*_i is corrupted due to the winning condition of $\mathsf{ExpEUFPARS}$, if $\mathsf{id}^*_i = \mathsf{id}^*$ for some $i \in [n]$, $\mathcal{A}$ can be used to break non-frameability of Π. Moreover, if $\mathsf{id}^*_i \neq \mathsf{id}^*$ for all $i \in [n]$ and $\mathsf{upk}_{\mathsf{id}^*} \notin \mathsf{L}_{\mathsf{CU}}$, then it violates traceability of Π. (More precisely, we can construct a PPT adversary $\mathcal{B}$ that breaks strong unforgeability of OTS, unforgeability of SIG, or simulation-sound extractability of ZK with non-negligible probability by following the same arguments of proofs of Lemma 1 or Lemma 2.)

However, even if $\mathsf{id}^*_i \neq \mathsf{id}^*$ for all $i \in [n]$ and $\mathsf{upk}_{\mathsf{id}^*} \in \mathsf{L}_{\mathsf{CU}}$, it contradicts the simulation-sound extractability of ZK. This is because c^* is an encryption of id^* that is not contained in the ring R^*, but π^* is a valid proof with respect to R^*. By a similar discussion as the proof of Lemma 2, if this case happens with

non-negligible probability, then we can construct a PPT adversary that breaks simulation-sound extractability of ZK with non-negligible probability.

To summarize the above discussion, we conclude that $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$ is negligible. Thus, Π is unforgeable if OTS is strongly unforgeable, SIG is unforgeable, and ZK is zero-knowledge and simulation-sound extractable. □

Lemma 4. *The construction Π is honest-OA anonymous if* PKE *is IND-CCA2 secure, and* ZK *is zero-knowledge.*

Proof. Let ϵ_{CCA2} be the advantage of PKE in the IND-CCA2 game, and ϵ_{ZK} be the advantage of ZK in the zero-knowledge game.

We let the first output by an adversary in $\mathsf{ExpHAnon}$ be as follows; $\mathsf{R}^* = (\mathsf{upk}_i^*)_{i\in[n]} = (\mathsf{id}_i^*, \mathsf{vk}_i^*)_{i\in[n]}$ be a ring, $\mathsf{usk}_{i_0}^* = (\mathsf{sigk}_{i_0}^*, \mathsf{cert}_{i_0}^*)$ and $\mathsf{usk}_{i_1}^* = (\mathsf{sigk}_{i_1}^*, \mathsf{cert}_{i_1}^*)$ be secret keys, and m^* be a message. Further, we assume that $\mathsf{upk}_{i_0}^* = (\mathsf{id}_{i_0}^*, \mathsf{vk}_{i_0}^*)$ and $\mathsf{upk}_{i_1}^* = (\mathsf{id}_{i_1}^*, \mathsf{vk}_{i_1}^*)$ can be retrieved from the lists, and they are contained in R^*.

We prove this lemma by a hybrid argument. Let P_i for $i \in \{0, 1, 2, 3\}$ be the probability that Hybrid i outputs 1 that are defined below.

Hybrid 0. We consider the case that the challenge signature is created by using $\mathsf{usk}_{i_0}^*$ (i.e., $b = 0$ in the experiment). Further, we assume that $f(\mathsf{id}_{i_0}^*) = f(\mathsf{id}_{i_1}^*) = 0$. In such a case, the challenger runs $\Pi.\mathsf{RSig}^{\mathsf{T}}$, and particularly, $\mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{OA}}, \mathsf{id}_{i_0}^*; \mathsf{r}_{\mathsf{PKE}})$ is executed for a randomness $\mathsf{r}_{\mathsf{PKE}}$ in its procedure. We omit the discussion for other combinations of $(f(\mathsf{id}_{i_0}^*), f(\mathsf{id}_{i_1}^*))$ because their proofs are somewhat incremental.

Hybrid 1. We replace $\mathsf{ZK.CRS}$ in the setup phase with $\mathsf{ZK.SimCRS}$, and $\mathsf{ZK.Prv}$ in the challenge phase with $\mathsf{ZK.Sim}$, respectively. In more detail, suppose that the challenger runs $\mathsf{ZK.Prv}(\mathsf{crs}, X, W)$ where $X = (f, \mathsf{R}^*, \mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, c^*, \mathsf{ovk}^*)$ and $W = ((\mathsf{id}_{i_0}^*, \mathsf{vk}_{i_0}^*), \mathsf{cert}_{i_0}^*, \sigma_{\mathsf{SIG}}, \mathsf{r}_{\mathsf{PKE}})$, in the challenge phase. In this hybrid, the challenger runs $(\mathsf{crs}, \tau) \leftarrow \mathsf{ZK.SimCRS}(1^\lambda)$ in the setup phase, and $\pi \leftarrow \mathsf{ZK.Sim}(\mathsf{crs}, X, \tau)$ in the challenge phase.

If an adversary $\mathcal{A}$ in $\mathsf{ExpHAnon}$ behaves differently due to this modification from Hybrid 0, it indicates that $\mathcal{A}$ breaks the zero-knowledge property of ZK. Thus, we have $|P_0 - P_1| \leq \epsilon_{\mathsf{ZK}}$.

Hybrid 2. Replace $\mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{OA}}, \mathsf{id}_{i_0}^*; \mathsf{r}_{\mathsf{PKE}})$ in the execution of $\Pi.\mathsf{RSig}^{\mathsf{T}}$ in the challenge phase with $\mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{OA}}, \mathsf{id}_{i_1}^*; \mathsf{r}_{\mathsf{PKE}})$. If an adversary $\mathcal{A}$ in $\mathsf{ExpHAnon}$ behaves differently due to this modification, it indicates that $\mathcal{A}$ breaks the IND-CCA2 security of PKE. Thus, we have $|P_1 - P_2| \leq \epsilon_{\mathsf{CCA2}}$.

Hybrid 3. We replace $\mathsf{ZK.SimCRS}$ in the setup phase with $\mathsf{ZK.CRS}$, and $\mathsf{ZK.Sim}$ in the challenge phase with $\mathsf{ZK.Prv}$, respectively. Particularly, the challenger runs $\mathsf{ZK.Prv}(\mathsf{crs}, X, W)$ for $X = (f, \mathsf{R}^*, \mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, c^*, \mathsf{ovk}^*)$ and $W = ((\mathsf{id}_{i_1}^*, \mathsf{vk}_{i_1}^*), \mathsf{cert}_{i_1}^*, \sigma_{\mathsf{SIG}}, \mathsf{r}_{\mathsf{PKE}})$, in the challenge phase. With the discussion similar to Hybrid 1, we have $|P_3 - P_2| \leq \epsilon_{\mathsf{ZK}}$.

Analysis. Observe that Hybrid 3 corresponds to the case where the challenger creates a signature for $\mathsf{usk}^*_{i_1}$. Therefore, $|P_3 - P_0|$ corresponds to the advantage of $\mathsf{ExpHAnon}$ (resp., $\mathsf{ExpCAnon}$). This value is at most $2\epsilon_{\mathsf{ZK}} + \epsilon_{\mathsf{CCA2}}$, which is negligible. □

We next show corrupted-OA anonymity of Π, but it can be proven in almost the same manner as honest-OA anonymity. The difference is that, in $\mathsf{ExpCAnon}$, $\mathsf{PKE.Enc}$ always encrypts 0 in the execution of $\Pi.\mathsf{RSig}^{\mathsf{A}}$ in the challenge phase. Thus, we can skip Hybrid 2. The remaining hybrids are the same, and thus, we conclude the lemma. We omit the details of the proof and provide only the statement of the lemma.

Lemma 5. *The construction Π is corrupted-OA anonymous if* PKE *is IND-CCA2 secure, and* ZK *is zero-knowledge.*

Finally, we demonstrate traceability of Π.

Lemma 6. *The construction Π is traceable if* SIG *is unforgeable,* OTS *is strongly unforgeable, and* ZK *is zero-knowledge and simulation-sound extractable.*

Proof. Traceability means that no PPT adversary is able to output a valid signature σ^* along with a ring R^* and a message m^* so that it opens to 0, without knowing any anonymous secret key. We prove this lemma using an argument similar to that in the proof of Lemma 1. That is, we let the challenger in ExpTrc use the extraction algorithm of ZK, and show that the probability that the extraction fails is negligible. However, if the extraction succeeds, then it means that it extracts a witness, namely valid signatures for which the adversary does not know the corresponding secret keys. This means that we can use the adversary to break unforgeability of SIG or strong unforgeability of OTS (by a similar discussion with the proof of Lemma 1). The formal description is provided below.

Hybrid 0. The ExpTrc game between the challenger and a PPT adversary $\mathcal{A}$. In the setup phase, the challenger first chooses a function $f \in \mathcal{F}$, initializes the lists $\mathsf{reg}, \mathsf{L}_{\mathsf{KG}}, \mathsf{L}_{\mathsf{HU}}, \mathsf{L}_{\mathsf{CU}}, \mathsf{L}_{\mathsf{SK}}, \mathsf{L}^{\mathsf{A}}_{\mathsf{Sig}}$, and $\mathsf{L}^{\mathsf{T}}_{\mathsf{Sig}}$ as empty sets. Next, it computes $(\mathsf{sk}_{\mathsf{OA}}, \mathsf{pk}_{\mathsf{OA}}) \leftarrow \mathsf{PKE.KG}(1^\lambda), (\mathsf{sigk}_{\mathsf{GM}}, \mathsf{vk}_{\mathsf{GM}}) \leftarrow \mathsf{SIG.KG}(1^\lambda)$, and $\mathsf{crs} \leftarrow \mathsf{ZK.CRS}(1^\lambda)$, sets $\mathsf{pp} := (f, \mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, \mathsf{crs}), \mathsf{msk} := \mathsf{sigk}_{\mathsf{GM}}$, and $\mathsf{oak} := \mathsf{sk}_{\mathsf{OA}}$, and provides pp to the adversary $\mathcal{A}$.

When $\mathcal{A}$ makes an oracle query, the challenger returns the answer. We only describe how $\mathsf{O}^{\mathsf{T}}_{\mathsf{GKG}}$ works, as the remaining parts are the same as Hybrid 0 in the proof of Lemma 1. (Furthermore, recall that $\mathsf{O}_{\mathsf{UKG}}$ is not provided in ExpTrc.)

$\mathsf{O}^{\mathsf{T}}_{\mathsf{GKG}}$**:** The adversary first sends $(\mathsf{id}, \mathsf{vk})$ to the oracle. Return $\perp$ if $f(\mathsf{id}) = 1$ or $\mathsf{upk} = (\mathsf{id}, \cdot) \in \mathsf{L}_{\mathsf{KG}} \cup \mathsf{L}_{\mathsf{HU}} \cup \mathsf{L}_{\mathsf{CU}}$. Otherwise, the oracle creates $\mathsf{cert} \leftarrow \mathsf{SIG.Sig}(\mathsf{sigk}_{\mathsf{GM}}, \mathsf{id}||\mathsf{vk}||0)$, and checks if $\mathsf{SIG.Vrf}(\mathsf{vk}_{\mathsf{GM}}, \mathsf{id}||\mathsf{vk}||0, \mathsf{cert}) = 1$. If the verification fails, then it outputs $\perp$. Otherwise, it sets $\mathsf{trans} := (\mathsf{id}, \mathsf{vk}, \mathsf{cert}), \mathsf{reg} := \mathsf{reg} \cup \{\mathsf{trans}\}$ and $\mathsf{L}_{\mathsf{CU}} := \mathsf{L}_{\mathsf{CU}} \cup \{\mathsf{upk}\}$, and outputs trans.

The other oracles are the same as those in the proof of Lemma 1.

Let $(\mathsf{R}^* = (\mathsf{upk}_i^*)_{i\in[n]}, \mathsf{m}^*, \sigma^*)$ be the output of $\mathcal{A}$. The challenger outputs 1 if $\forall \mathsf{upk}_{\mathsf{id}} \in \mathsf{L}_{\mathsf{CU}}$ such that $f(\mathsf{id}) = 1$, $(\mathsf{upk}_{\mathsf{id}}, \mathsf{R}^*, \mathsf{m}^*, 1, \sigma^*) \notin \mathsf{L}^{\mathsf{A}}_{\mathsf{Sig}}$, $\mathit{\Pi}.\mathsf{Vrf}(\mathsf{pp}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) = 1$ and $\mathit{\Pi}.\mathsf{Open}(\mathsf{pp}, \mathsf{oak}, \mathsf{R}^*, \mathsf{m}^*, \sigma^*) = 0$, otherwise 0.

Hybrid 1. Modify Hybrid 0 so that the challenger runs $(\mathsf{crs}, \tau, \zeta) \leftarrow \mathsf{ZK.ExtCRS}(1^\lambda)$ instead of $\mathsf{ZK.CRS}$ in the setup phase. Further, we let the challenger run $\mathsf{ZK.Ext}(\mathsf{crs}, \zeta, X^*, \pi^*)$ just before outputting the final bit.

Similarly to the proof of Lemma 1, we define P_i for $i \in \{0, 1\}$, Succ_i for $i \in \{0, 1\}$ and Bad. We have that $P_0 \leq |P_1 - P_0| + P_1$, $|P_1 - P_0|$ is negligible, $P_1 = \Pr[\mathsf{Succ}_1 \wedge \mathsf{Bad}] + \Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$, and $\Pr[\mathsf{Succ}_1 \wedge \mathsf{Bad}]$ is negligible, due to the same discussion as the proof of Lemma 1.

Now, we evaluate $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$. If $\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}$ happens, it means that $\mathcal{A}$ wins the game and $\mathsf{ZK.Ext}$ extracts a correct witness. As $\mathcal{A}$ wins the game, the ciphertext in the forgery is an encryption of 0. Furthermore, since $\mathsf{ZK.Ext}$ extracts a correct witness, the challenger obtains $\mathsf{upk}_{\mathsf{id}^*} = (\mathsf{id}^*, \mathsf{vk}_{\mathsf{id}^*})$ such that $f(\mathsf{id}^*) = 1$, $\mathsf{cert}_{\mathsf{id}^*}$ such that $\mathsf{SIG.Vrf}(\mathsf{vk}_{\mathsf{GM}}, \mathsf{id}^*||\mathsf{vk}_{\mathsf{id}^*}||f(\mathsf{id}^*), \mathsf{cert}_{\mathsf{id}^*}) = 1$, and σ^*_{SIG} such that $\mathsf{SIG.Vrf}(\mathsf{vk}_{\mathsf{id}^*}, \mathsf{ovk}^*, \sigma^*_{\mathsf{SIG}}) = 1$ (note that the challenger also obtains the randomness $\mathsf{r}_{\mathsf{PKE}}$, but we omit it here because it is not used in our proof).

Before showing reduction algorithms, we discuss how $\mathsf{upk}_{\mathsf{id}^*}$ is generated. In the proof of Lemma 1, the target key was created via oracles due to the winning condition of $\mathsf{ExpNFrm}$ (i.e., $\mathsf{upk}^* \in \mathsf{L}_{\mathsf{KG}} \cup \mathsf{L}_{\mathsf{HU}}$). However, as no such restriction is given in ExpTrc, we should consider how $\mathcal{A}$ obtains $\mathsf{upk}_{\mathsf{id}^*}$. Observe that $\mathsf{upk}_{\mathsf{id}^*}$ cannot be generated by $\mathsf{O}^{\mathsf{T}}_{\mathsf{GKG}}$, as it returns $\perp$ when given id such that $f(\mathsf{id}) = 1$. Thus, $\mathcal{A}$ obtains $\mathsf{upk}_{\mathsf{id}^*}$ via O_{KG}, or without using key generation oracles.

We first consider the latter case. In this case, the challenger obtains $\mathsf{cert}_{\mathsf{id}^*}$ by the extraction, without running O_{KG}. Thus, in this case, we can construct a reduction algorithm $\mathcal{B}$ that breaks unforgeability of SIG as follows.

Setup Phase. Given vk from the challenger of $\mathsf{ExpEUFSig}$, $\mathcal{B}$ sets up ExpTrc as Hybrid 1, except that it uses vk as $\mathsf{vk}_{\mathsf{GM}}$ instead of producing it.

Forgery Phase. When $\mathcal{A}$ makes oracle queries, $\mathcal{B}$ answers in the same manner as Hybrid 1. As $\mathcal{B}$ does not know the signing key corresponding to vk, $\mathcal{B}$ should call the signing oracle in $\mathsf{ExpEUFSig}$ in the following oracle simulations: Namely, in the simulation of O_{KG} and $\mathsf{O}^{\mathsf{T}}_{\mathsf{GKG}}$, when $\mathcal{B}$ needs to create a certificate for id, $\mathcal{B}$ calls the signing oracle on $\mathsf{id}||\mathsf{vk}||f(\mathsf{id})$ to receive the certificate. Note that in the simulation of $\mathsf{O}^{\mathsf{T}}_{\mathsf{GKG}}$, $\mathcal{B}$ returns $\perp$ given id such that $f(\mathsf{id}) = 1$.

When $\mathcal{A}$ outputs $\sigma^* = (\mathsf{ovk}^*, c^*, \pi^*, \sigma^*_{\mathsf{OTS}})$, $\mathcal{B}$ runs $\mathsf{ZK.Ext}$ to extract a witness. Particularly, $\mathcal{B}$ obtains the signature $\mathsf{cert}_{\mathsf{id}^*}$ on $\mathsf{id}^*||\mathsf{vk}^*||1$, and outputs $\mathsf{cert}_{\mathsf{id}^*}$ and $\mathsf{id}^*||\mathsf{vk}^*||1$.

Analysis. Recall that we are considering the case that $\mathsf{upk}_{\mathsf{id}^*}$ is not created by oracles. Thus, $\mathsf{cert}_{\mathsf{id}^*}$ is a fresh signature for $\mathcal{B}$ on the message $\mathsf{id}^*||\mathsf{vk}^*||1$.

Therefore, if $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$ is non-negligible, then $\mathcal{B}$ breaks unforgeability of SIG with non-negligible probability.

Finally, we consider the case that $\mathsf{upk}_{\mathsf{id}^*}$ is created by O_{KG}. In this case, we cannot regard $\mathsf{cert}_{\mathsf{id}^*}$ as a forgery. That is, suppose that we construct the above reduction algorithm for this case. As $\mathsf{upk}_{\mathsf{id}^*}$ is created by O_{KG}, the reduction algorithm $\mathcal{B}$ receives $\mathsf{cert}_{\mathsf{id}^*}$ via the signing oracle of $\mathsf{ExpEUFSig}$. Thus, $\mathsf{cert}_{\mathsf{id}^*}$ is no longer a fresh signature.

We recall that the extraction algorithm extracts the signature σ^*_{SIG} such that $\mathsf{SIG.Vrf}(\mathsf{vk}_{\mathsf{id}^*}, \mathsf{ovk}^*, \sigma^*_{\mathsf{SIG}}) = 1$. Thus, we consider constructing reduction algorithms that output σ^*_{SIG} as a forgery. The remaining part of this proof is almost the same as that of Lemma 1. That is, we consider two cases; (i) ovk^* is not queried to the signing oracle, and (ii) ovk^* is queried. In the former case, we show that, if $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$ is non-negligible, then we can construct a PPT reduction algorithm that breaks unforgeability of SIG. In the latter case, we show that, if $\Pr[\mathsf{Succ}_1 \wedge \neg\mathsf{Bad}]$ is non-negligible, then we can construct a PPT reduction algorithm that breaks strong unforgeability of OTS. In conclusion, the construction Π is traceable if SIG is unforgeable, OTS is strongly unforgeable, and ZK is zero-knowledge and simulation-sound extractable. □

4.3 Discussion on Further Extensions of PARS

We conclude this section by discussing two possible extensions of our PARS framework. These extensions are not part of the basic construction analyzed above, but they illustrate the flexibility of the PARS approach and clarify some limitations of the present model.

First, our construction can be extended so that the opening algorithm reveals information more flexibly. In the basic construction, opening reveals the signer's identity for traceable signatures and returns the special symbol 0 for anonymous signatures. More generally, one may wish to reveal only selected information about the signer, such as an attribute, a role, or an authority level. This can be achieved by modifying the statement and witness relation of the NIZKPoK so that the ciphertext encrypts the prescribed information determined by a function of the signer's identity or attributes.

Second, one may consider anonymity in a malicious-setup model. Our current definitions assume that the setup algorithm is honestly executed. This is because, if the CRS of the NIZKPoK were generated maliciously, it might contain hidden information that enables the setup authority to distinguish the witness used in a proof, thereby undermining signer anonymity. Protecting anonymity in this setting appears to require stronger tools, such as subversion-resistant NIZK [2].

More Flexible Opening. As discussed in Section 1.2, we can extend PARS to support more flexible opening. Let $f : \mathcal{ID} \to [I]$ be a function for a given integer I. Moreover, define another function $g : [I] \to [I']$ for an integer I'. The function g specifies what information is disclosed by the OA upon opening. For example,

$g(f(\mathsf{id})) = 2$ implies that the OA can only disclose the professional position of the user id. By using a NIZKPoK for the following language, we can achieve PARS with stepwise disclosure of signer information (i.e., more personalized PARS) like multimodal private signatures [23].

$$\begin{aligned}\mathcal{L} =\{&\mathsf{R}, \mathsf{vk}_{\mathsf{GM}}, \mathsf{pk}_{\mathsf{OA}}, c, \mathsf{ovk} \mid \exists \mathsf{upk}_{\mathsf{id}^*} = (\mathsf{id}^*, \mathsf{vk}_{\mathsf{id}^*}) \in \mathsf{R}, \mathsf{cert}_{\mathsf{id}^*}, \sigma_{\mathsf{SIG}}, \mathsf{r}_{\mathsf{PKE}} \text{ s.t.} \\ &\mathsf{SIG.Vrf}(\mathsf{vk}_{\mathsf{GM}}, \mathsf{id}^* || \mathsf{vk}_{\mathsf{id}^*} || f(\mathsf{id}^*), \mathsf{cert}_{\mathsf{id}^*}) = 1 \wedge \mathsf{SIG.Vrf}(\mathsf{vk}_{\mathsf{id}^*}, \mathsf{ovk}, \sigma_{\mathsf{SIG}}) = 1 \\ &\wedge (\exists i \in [I] \text{ s.t. } f(\mathsf{id}^*) = i \wedge c = \mathsf{PKE.Enc}(\mathsf{pk}_{\mathsf{OA}}, g(f(\mathsf{id}^*)); \mathsf{r}_{\mathsf{PKE}}))\}\end{aligned}$$

For such a PARS construction, the security definitions might require further refinement. However, we believe that the essential formalization has already been achieved in this paper. Furthermore, we emphasize that this paper primarily focuses on decoupling anonymity and traceability for each user, which we consider to be the most crucial perspective.

On the Definition of Anonymity. As mentioned in Section 3, we might be able to define $\mathsf{ExpHAnon}$ and $\mathsf{ExpCAnon}$ so that the setup algorithm is run maliciously. Even under such circumstances, there may still be a way to ensure the security of our PARS.

The key ingredient is subversion-resistant NIZK [2]. Roughly speaking, a subversion-resistant NIZK is an NIZK in which the witness cannot be learned from the proof, even if the CRS is generated maliciously. Therefore, if we adopt a subversion-resistant NIZK in our construction, it could maintain anonymity even though the setup algorithm is run maliciously.

5 Conclusion and Future Work

In this paper, we proposed a new primitive called personalized-anonymity ring signature (PARS), which allows for flexible switching between anonymity and traceability depending on factors such as user roles. We provided a generic construction of PARS using standard cryptographic building blocks: (standard) signatures, one-time signatures, PKE, and NIZKPoK. PARS is particularly effective for applications like internal whistleblowing, where lower-ranking employees require anonymity for protection, while higher-ranking officials must be held accountable for their statements.

A primary direction for future work is to incorporate more versatile traceability features into PARS. Specifically, it is an important challenge to enable switching between anonymity and traceability based on the message being signed (similar to bifurcated anonymous signatures [18]).

Another promising direction is the application of personalized anonymity to anonymous signatures other than ring signatures. For instance, Multi-Designated Verifier Signatures (MDVS) are a strong candidate, as they inherently ensure signer anonymity. Note that it has been recently shown that MDVS cannot be achieved from ring signatures in a black-box manner [32].

Acknowledgement

The authors declare that they used an AI tool, GPT-5.5, in preparing this manuscript. The tool was used solely for English proofreading. The selection of the research topic, the development of the formalizations and proofs, and the construction of the paper's storyline were carried out entirely by the authors.

References

1. Bellare, M., Fuchsbauer, G.: Policy-based signatures. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 520–537. Springer, Berlin, Heidelberg (Mar 2014). DOI: 10.1007/978-3-642-54631-0_30
2. Bellare, M., Fuchsbauer, G., Scafuro, A.: NIZKs with an untrusted CRS: Security in the face of parameter subversion. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part II. LNCS, vol. 10032, pp. 777–804. Springer, Berlin, Heidelberg (Dec 2016). DOI: 10.1007/978-3-662-53890-6_26
3. Bellare, M., Micciancio, D., Warinschi, B.: Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 614–629. Springer, Berlin, Heidelberg (May 2003). DOI: 10.1007/3-540-39200-9_38
4. Bellare, M., Shi, H., Zhang, C.: Foundations of group signatures: The case of dynamic groups. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 136–153. Springer, Berlin, Heidelberg (Feb 2005). DOI: 10.1007/978-3-540-30574-3_11
5. Bender, A., Katz, J., Morselli, R.: Ring signatures: Stronger definitions, and constructions without random oracles. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 60–79. Springer, Berlin, Heidelberg (Mar 2006). DOI: 10.1007/11681878_4
6. Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J., Petit, C.: Short accountable ring signatures based on ddh. In: Computer Security – ESORICS 2015: 20th European Symposium on Research in Computer Security, Vienna, Austria, September 21-25, 2015, Proceedings, Part I. p. 243–265. Springer-Verlag, Berlin, Heidelberg (2015). DOI: 10.1007/978-3-319-24174-6_13, https://doi.org/10.1007/978-3-319-24174-6_13
7. Champion, J., Hsieh, Y.C., Wu, D.J.: Registered ABE and adaptively-secure broadcast encryption from succinct LWE. In: Kalai, Y.T., Kamara, S.F. (eds.) CRYPTO 2025, Part III. LNCS, vol. 16002, pp. 3–34. Springer, Cham (Aug 2025). DOI: 10.1007/978-3-032-01881-6_1
8. Chaum, D., van Heyst, E.: Group signatures. In: International Conference on the Theory and Application of Cryptographic Techniques (1991), https://api.semanticscholar.org/CorpusID:970624
9. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO'98. LNCS, vol. 1462, pp. 13–25. Springer, Berlin, Heidelberg (Aug 1998). DOI: 10.1007/BFb0055717
10. De Santis, A., Di Crescenzo, G., Ostrovsky, R., Persiano, G., Sahai, A.: Robust non-interactive zero knowledge. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 566–598. Springer, Berlin, Heidelberg (Aug 2001). DOI: 10.1007/3-540-44647-8_33
11. Fujisaki, E., Suzuki, K.: Traceable ring signature. Cryptology ePrint Archive, Report 2006/389 (2006), https://eprint.iacr.org/2006/389

12. Ghadafi, E.: Efficient distributed tag-based encryption and its application to group signatures with efficient distributed traceability. In: Aranha, D.F., Menezes, A. (eds.) Progress in Cryptology - LATINCRYPT 2014. pp. 327–347. Springer International Publishing, Cham (2015)
13. Groth, J.: Simulation-sound NIZK proofs for a practical language and constant size group signatures. In: Lai, X., Chen, K. (eds.) ASIACRYPT 2006. LNCS, vol. 4284, pp. 444–459. Springer, Berlin, Heidelberg (Dec 2006). DOI: 10.1007/11935230_29
14. Hwang, J.Y., Chen, L., Cho, H.S., Nyang, D.: Short dynamic group signature scheme supporting controllable linkability. IEEE Transactions on Information Forensics and Security **10**(6), 1109–1124 (2015). DOI: 10.1109/TIFS.2015.2390497
15. Kiayias, A., Tsiounis, Y., Yung, M.: Traceable signatures. Cryptology ePrint Archive, Report 2004/007 (2004), https://eprint.iacr.org/2004/007
16. Kohlweiss, M., Miers, I.: Accountable tracing signatures. Cryptology ePrint Archive, Paper 2014/824 (2014), https://eprint.iacr.org/2014/824
17. Komano, Y., Ohta, K., Shimbo, A., Kawamura, S.: Toward the fair anonymous signatures: Deniable ring signatures. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 174–191. Springer, Berlin, Heidelberg (Feb 2006). DOI: 10.1007/11605805_12
18. Libert, B., Nguyen, K., Peters, T., Yung, M.: Bifurcated signatures: Folding the accountability vs. anonymity dilemma into a single private signing scheme. In: Canteaut, A., Standaert, F.X. (eds.) Advances in Cryptology – EUROCRYPT 2021. pp. 521–552. Springer International Publishing, Cham (2021)
19. Ling, S., Nguyen, K., Wang, H., Xu, Y.: Accountable tracing signatures from lattices. In: Matsui, M. (ed.) CT-RSA 2019. LNCS, vol. 11405, pp. 556–576. Springer, Cham (Mar 2019). DOI: 10.1007/978-3-030-12612-4_28
20. Liu, J.K., Wong, D.S.: Linkable ring signatures: Security models and new schemes. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganà, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) Computational Science and Its Applications – ICCSA 2005. pp. 614–623. Springer Berlin Heidelberg, Berlin, Heidelberg (2005)
21. Maji, H., Prabhakaran, M., Rosulek, M.: Attribute-based signatures: Achieving attribute-privacy and collusion-resistance. Cryptology ePrint Archive, Report 2008/328 (2008), https://eprint.iacr.org/2008/328
22. Manulis, M.: Democratic group signatures: on an example of joint ventures. In: Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security. p. 365. ASIACCS '06, Association for Computing Machinery, New York, NY, USA (2006). DOI: 10.1145/1128817.1128882, https://doi.org/10.1145/1128817.1128882
23. Nguyen, K., Guo, F., Susilo, W., Yang, G.: Multimodal private signatures. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology – CRYPTO 2022. pp. 792–822. Springer Nature Switzerland, Cham (2022)
24. Park, S., Sealfon, A.: It wasn't me! - Repudiability and claimability of ring signatures. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 159–190. Springer, Cham (Aug 2019). DOI: 10.1007/978-3-030-26954-8_6
25. Peikert, C., Shiehian, S.: Noninteractive zero knowledge for NP from (plain) learning with errors. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part I. LNCS, vol. 11692, pp. 89–114. Springer, Cham (Aug 2019). DOI: 10.1007/978-3-030-26948-7_4
26. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 187–196. ACM Press (May 2008). DOI: 10.1145/1374376.1374406

27. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Berlin, Heidelberg (Dec 2001). DOI: 10.1007/3-540-45682-1_32
28. Sakai, Y., Emura, K., Hanaoka, G., Kawai, Y., Matsuda, T., Omote, K.: Group signatures with message-dependent opening. In: Abdalla, M., Lange, T. (eds.) PAIRING 2012. LNCS, vol. 7708, pp. 270–294. Springer, Berlin, Heidelberg (May 2013). DOI: 10.1007/978-3-642-36334-4_18
29. Shaw, S., Dutta, R.: Key-oblivious encryption from isogenies with application to accountable tracing signatures. In: Adhikari, A., Küsters, R., Preneel, B. (eds.) INDOCRYPT 2021. LNCS, vol. 13143, pp. 362–386. Springer, Cham (Dec 2021). DOI: 10.1007/978-3-030-92518-5_17
30. Wei, V.K.: Tracing-by-linking group signatures. In: Zhou, J., Lopez, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 149–163. Springer, Berlin, Heidelberg (Sep 2005). DOI: 10.1007/11556992_11
31. Xu, S., Yung, M.: Accountable ring signatures: A smart card approach. In: Quisquater, J.J., Paradinas, P., Deswarte, Y., El Kalam, A.A. (eds.) Smart Card Research and Advanced Applications VI. pp. 271–286. Springer US, Boston, MA (2004)
32. Yamashita, K., Hara, K.: On the black-box impossibility of multi-designated verifiers signature schemes from ring signature schemes. Journal of Mathematical Cryptology **18**(1), 20230028 (2024). DOI: doi:10.1515/jmc-2023-0028, https://doi.org/10.1515/jmc-2023-0028