IACR Communications in Cryptology
ISSN 3006-5496, Vol. 1, No. 3, 36 pages.

https://doi.org/10.62056/a33zivrzn

Multi Designated Verifier Ring Signatures

Sebastian Kolby[1], Elena Pagnin[2] and Sophia Yakoubov[1]

[1] Aarhus University, Computer Science, Aarhus, Denmark
[2] Chalmers University of Technology and University of Gothenburg, Computer Science and Engineering, Gothenburg, Sweden

Abstract.
We study signatures well suited for sensitive applications (e.g. whistleblowing) where both the signer's anonymity and deniability are important. Two independent lines of work have tackled these two goals: *ring signatures* ensure the signer's anonymity (within a set of signers, called a ring), and — separately — *multi designated verifier signatures* ensure that all the intended recipients agree on whether a signature is valid, while maintaining the signer's deniability by preventing the intended recipients from convincing an outsider of the validity of the signature. In this paper, we introduce *multi designated verifier ring signatures* (MDVRS), which simultaneously offer both signer anonymity and deniability. This makes MDVRS uniquely suited for sensitive scenarios.
Following the blueprint of Damgård et al (TCC'20) for multi designated verifier signatures, we introduce *provably simulatable designated verifier ring signatures* (PSDVRS) as an intermediate building block which we then compile into an MDVRS. We instantiate PSDVRS in a concretely efficient way from discrete logarithm based sigma protocols, encryption and commitments.

Keywords: Digital Signatures · Designated Verifier · Ring Signatures.

1 Introduction

Digital signatures serve a fundamental role in securing online communication. A digital signature scheme enables the signer, i.e., the party holding a secret key, to produce a concise string, called a signature, for a chosen message. This signature authenticates the chosen message in such a way that any verifier, i.e., any party holding the signer's public key, can check that the signature was produced by the signer and that the message was not altered. This fairly simple property is integral to countless applications, including digital certificates, electronic payment systems, and identification schemes.

While the initial aim of digital signatures was to make the origin of a message publicly verifiable, over the years interest has grown towards authenticating messages in a *privacy-enhancing* way. This can serve two purposes:

1. Protecting the signer by (1) hiding their identity within an *anonymity set* or (2) making it harder for the intended verifier to implicate them. This is relevant when the signer is a whistle-blower and might face retaliation, and may address the privacy threat associated with dissemination of signed documents [SBWP03].

Funded in part by the Danish Independent Research Council (grants DFF-2064-00016B and DFF-2032-00122B "YOSO", and grant DFF-0165-00107B "C3PO"), and the European Research Council (ERC, under the European Unions's Horizon 2020 research and innovation programme under grant agreement No 803096 "SPEC").

E-mail: sk@cs.au.dk (Sebastian Kolby), elenap@chalmers.se (Elena Pagnin), sophia.yakoubov@cs.au.dk (Sophia Yakoubov)

Received: 2024-07-08 Accepted: 2024-09-02

2. Controlling exposure by proving authenticity only to preselected — authorized — entities. This provides new venues of application for digital signatures, such as verifying copies of software only for costumers who paid [JSI96], and in blockchain technologies [CK21].

A signer's privacy can be protected in several ways. *Ring signatures* (RS) [RST01] and *group signatures* [Cv91] both hide the signer's exact identity among a set of potential signers. In the case where the signer is an informant or whistle-blower, this has the drawback of creating evidence that at least one member of a group leaked information; while the exact identity of the group member remains unknown, such evidence might trigger an investigation or mass mistrust. This is particularly problematic when the group (ring) is small. To combat this, Aranha et al. [AHAN+22] propose a way to extend the initial ring to a larger set; however, the public verifiability issue remains.

In contrast, *designated verifier* [JSI96] signatures limit who is able to perform the verification of message authenticity. A traditional signature can be verified by anyone, but a designated verifier signature only convinces the intended recipient of the signature's authenticity. That recipient is then unable to prove to anyone else that the signature is authentic. *Multi designated verifier* signatures (MDVS) [LV04, DHM+20] allow the signer to enable verification by more than one recipient (while still limiting the recipients' ability to convince an outsider). As long as no retaliator is among the designated verifiers, the retaliators have no evidence implicating an individual or group. However, given that each designated verifier is convinced of the exact identity of the signer, signers are still at risk in case one of the designated verifiers happened to be a retaliator, or have a retaliator's trust.

The two approaches have been recently combined into *designated verifier linkable ring signatures* [BBG+22]. However, the case of *multi* designated verifier ring signatures — which additionally guarantee consistent verification by multiple recipients — remains unexplored. In this work, we are interested in filling this gap in privacy-oriented signatures. At a first glance, this may appear as a trivial combination of ring signatures and multi designated verifier techniques; however, it turns out that simultaneously achieving both the *signer-ring* feature and the *multi* designated verifier one is far from trivial.

Properties Multi designated verifier ring signatures (MDVRS), which we introduce, provide stronger security notions for signer privacy by (1) protecting the signer's identity within a ring of potential signers, and (2) limiting the evidence that could be used against them by selecting a set of designated verifiers. On top of *correctness* and *unforgeability*, MDVRS signatures enjoy:

Consistency, i.e., all designated verifiers get the same validity result.

Off-the-record (a.k.a. deniability), i.e., no set of colluding verifiers can provably identify a smaller ring of signers and prove this to a third party.

Anonymity, i.e., the identity of the actual signer is not revealed even in case all signers' and designated verifiers' secret keys involved in ring signature are known.

Additional Applications Whistleblowing remains the most compelling motivation for privacy-preserving signatures. Other applications include *e-voting* and *alarm-sounding*.

E-voting: Privacy-preserving signatures can enable the anonymous casting of a ballot. Proving authenticity only to a designated set of verifiers makes it harder for a coercer to check that a voter voted a certain way.

Sounding an alarm: It may be important to alert a localized group to a particular danger. The person sounding that alarm might not want to be identified, to avoid questioning. This alarm might, for instance, consist of notifying nearby hospitals of an outbreak of coronavirus.

1.1 Our Contribution

We put forth a definitional framework and a construction for *multi* designated verifier *ring* signatures (MDVRS). As a building block for MDVRS, and a result of independent interest, we extend the notion of provably simulatable designated verifier signatures [DHM+20] to the ring setting (PSDVRS). We give a generic compiler from PSDVRS to MDVRS, and we design a concretely efficient PSDVRS from discrete logarithm-based assumptions. Our PSDVRS uses a discrete logarithm-based non-interactive zero-knowledge proof system.

Finally, we discuss how our approach may be applied to a compressed sigma protocol [AC20] for proving partial knowledge (1-of-n) of discrete logarithms [ACF21] to achieve PSDVRS signatures which have size logarithmic in the ring.

The size of our resulting MDVRS signature has a linear dependency on the number of verifiers, and a logarithmic dependency on the signer ring. Notably, the linear dependency on the number of verifiers cannot be avoided, as shown by Damgård *et al.* [DHM+20] (in the context of MDVS).

1.2 Technical Overview

We build our MDVRS scheme in a generic way from a *provably simulatable designated (single) verifier ring signature (PSDVRS)* scheme. The idea is that a PSDVRS signature can be produced in one of three ways: by the signer, by the verifier, or by a third party (in which case it should not verify — since otherwise it would constitute a forgery — but it should look like a valid signature to an outsider). Furthermore, the signature is produced together with a proof of the *way* in which it was generated; this proof demonstrates that the signature is real, that it is a simulation by the designated verifier, or that it is a simulation by a third party. Without the proof, signatures produced in the three ways are indistinguishable, and signatures produced the first two ways are indistinguishable even given the verifier's secret state. In order to produce an MDVRS signature, the signer produces a PSDVRS signature for each verifier; to ensure consistency, the signer proves in zero knowledge that either *all* the signatures are real and for the same message, or they are *all* a kind of simulation.

Now, the challenge becomes building a PSDVRS signature that is both concretely efficient, and that lends itself to efficient zero knowledge OR-proofs of the kind described above. From a structural perspective our PSDVRS signature is based on a ring signature, where the ring is augmented with the verifier's own key to ensure the off-the-record property. For efficiency, we design our ring signature scheme based on a concrete Sigma protocol for the knowledge of one of several discrete logarithms. To embed the designated verifier feature, the signer encrypts the first message of the Sigma protocol to the verifier using ElGamal encryption in the same group where the keys live. This trick prevents public verifiability of the Sigma protocol and, by compiling the Sigma protocol into a non interactive honest verifier zero knoweldge (HVZK) argument of knowledge (Figure 17), also provides public simulatability. Indeed, the resulting signatures are publicly simulatable by producing a dummy ciphertext for the first message of the Sigma protocol, and then using the simulator that enables the honest verifier zero knowledge property of the (compiled) argument of knowledge. To distinguish the origin of simulated signatures it suffices to include the randomness associated to the dummy ciphertext in the proof of the signature's provenance. To prove provenance of signatures generated by the signer (resp. verifier), the signer (resp. verifier) includes a commitment to their own secret key in the signature; and provides its secret key and commitment randomness in the proof of provenance. Again, public simulation will commit to a dummy value (and include the associated randomness in the proof of provenance). We achieve this, while maintaining a non interactive zero knowledge (NIZK) friendly statement, avoiding proving the evaluation of cryptographic hashes within zero knowledge.

1.3 Related Work

Ring Signatures Ring signatures [RST01] are a well established mechanism to leak secrets in an anonymous way and yet show some entitlement [BGLS03, ZK02, DKNS04, BKM06]. The concept of ring signature has been revisited and enhanced over the years to include, e.g., threshold versions *et al.* [BSS02], traceability [FS07], linkability [SALY17], identity based constructions [CLHY05], and ring extensions [AHAN+22]. In this work, we are interested in combining it with the verifier's designation.

Designated Verifier Signatures The notion of designated verifier proofs was introduced in 1996 independently by Chaum (in a patent) and by Jakobsson, Sako and Impagliazzo [JSI96]. One major application for this primitive are designated verifier signatures (DVS) which provide deniability (aka off-the-record) and authentication at the same time, emulating the properties of a face-to-face conversation. (In a face-to-face conversation, authentication is given by witnessing the speaker make a statement, and off-the-record comes from later inability to prove what was said.) Among other things, DVS enables a software vendor to certify that products are correct and free of viruses, but only to paying customers [JSI96]. Jakobsson *et al.* [JSI96] use trapdoor (chameleon) commitments to build DVS.

In 2003, to address the privacy threat associated with dissemination of verifiable signed documents, Steinfeld, Bull, Wang and Pieprzyk [SBWP03] and [SWP04] generalize the concept of DVS to Universal DVS, which allows anyone (not necessarily the signer themselves) to transform a publicly verifiable signature into a designated verifier signature for any chosen verifier. The designated verifier can verify this signature and be convinced that the message was signed by the signer; but, they are unable to convince anyone else of this fact. Steinfeld *et al.* [SBWP03] provide constructions of deterministic Universal DVS based on BLS [BLS01] signatures (in the random oracle model).

Laguillaumie and Vergnaud introduce the concept of Multi Designated Verifiers Signatures (MDVS) [LV04], an extension of DVS. They identify core challenges for this setting, including consistency of the designated verifiers' verification outcomes, unforgeability even given all of the designated verifiers' keys, source hiding and privacy of signer's identity. Their constructions are based on ring signatures, or on bilinear maps (with tripartite Joux key exchange [Jou04]). This work sparked a extensive line of research that revisited the seminal constructions [Cho08], proposed identity based solutions [Cho06], and strengthened the security [DHM+20].

On the single verifier side, several works consider efficiency [SKM04], identity based key management systems [SZM04], a unified generic framework [LSMP07], but never signer privacy, until very recently. Recently, designated verifier linkable ring signatures [BGK+22] were introduced, to which Balla *et al.* [BBG+22] added unconditional anonymity[1]. These works combine signer anonymity with linkability, which makes same signer signatures traceable.

1.4 Comparison to Balla *et al.* [BBG+22]

Designated verifier linkable ring signatures [BBG+22] simultaneously offer the signer limited anonymity against the designated verifier, and deniability against everyone else. Anonymity is limited because linkability ensures that two signatures produced by the same signer can be linked (to one another, but not to the signer's identity). In particular, if a signer picks inappropriate rings (e.g., where the signer is the only entity in the intersection of the two rings) for the same 'ev', linkability will reveal the signer's identity.

[1] Unconditional anonymity states that the anonymity property does not rely on any computational assumptions (this holds as long as 'ev' was not reused).

In this paper, we propose a concrete construction that is based on that of Balla *et al.*, modified to remove the linkability tags and to achieve publicly simulatable designated verifier ring signatures. We then use this modified construction as a building block for our MDVRS.

2 Preliminaries

Here we briefly recall the main cryptographic primitives used as building blocks in our constructions.

Commitment Scheme. Let $\mathsf{Com} = (\mathsf{Setup}, \mathsf{Commit})$ be a non interactive commitment scheme that is hiding and binding [BFM88]. For efficiency reasons we consider the perfectly hiding and computationally binding commitments of [Ped92].

Non interactive zero knowledge argument of knowledge. Let the tuple $\mathsf{NIZK} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ be a non interactive zero knowledge argument of knowledge scheme. Taking the definitions of Groth and Maller [GM17] we will require perfect completeness, zero knowledge and simulation extractability. Note, simulation extractability is a stronger property than simulation soundness.

3 Multi Designated Verifier Ring Signatures

In this section, we introduce the notion of Multi Designated Verifier Ring Signatures (MDVRS) with syntax (Section 3.1) and security model (Section 3.2). Throughout the section we focus on the multi verifier case; the syntax for the basic case of a single designated verifier can be easily derived by setting $|\mathcal{D}| = 1$. Our construction of MDVRS is deferred to Section 5.

3.1 Syntax

We follow the approach of Damgård *et al.* [DHM+20] and include two key-generation algorithms in the syntax: one to be run for signers, the other for (designated) verifiers. This split enables a semantic distinction between these entities which is useful in real world scenarios where signers may not necessarily wish to act as verifiers, and vice versa.

$\mathtt{pp}$ is an implicit input to all algorithms after Setup.

For the sake of notation, σ denotes a signature on the message μ, for the ring of potential signers identified by $\mathcal{R}$, and for designated verifiers in $\mathcal{D}$.

Definition 1 (MDVRS)**.** A *multi designated verifier ring signature* (MDVRS) scheme is defined by the following probabilistic polynomial-time (PPT) algorithms:

$\mathsf{Setup}(1^\kappa) \to \mathtt{pp}$**:** On input the security parameter $\kappa \in \mathbb{N}$, outputs public parameters $\mathtt{pp}$ that are implicitly input to all subsequent algorithms and contain at least a description of the message, key, and user index spaces.

$\mathsf{SignKeyGen}() \to (\mathtt{spk}, \mathtt{ssk})$**:** On input the public parameter $\mathtt{pp}$, outputs the public key $\mathtt{spk}$ and secret key $\mathtt{ssk}$ for a signer.

$\mathsf{VerKeyGen}() \to (\mathtt{vpk}, \mathtt{vsk})$**:** On input the public parameter $\mathtt{pp}$, outputs the public key $\mathtt{vpk}$ and secret key $\mathtt{vsk}$ for a verifier.

$\mathsf{Sign}(\mathtt{ssk}_k, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}}, \mu) \to \sigma$**:** On input the public parameters $\mathtt{pp}$; a secret signing key $\mathtt{ssk}_k$ of the signer $k \in \mathcal{R}$ (this enforces that the signer's public key appears in the set $\{\mathtt{spk}_i\}_{i\in\mathcal{R}}$); a set $\{\mathtt{spk}_i\}_{i\in\mathcal{R}}$ of signers' public keys in the ring of

identifiers $\mathcal{R}$; a set $\{\mathtt{vpk}_j\}_{j\in\mathcal{D}}$ of designated verifiers' public keys; and a message μ; the algorithm outputs a signature σ.

$\mathsf{Verify}(\mathtt{vsk}_k, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}}, \mu, \sigma) = d \in \{0,1\}$**:** On input the public parameters $\mathtt{pp}$; a secret verifier key $\mathtt{vsk}_k$ for a designated verifier $k \in \mathcal{D}$; a set of public signers keys $\{\mathtt{spk}_i\}_{i\in\mathcal{R}}$; a set of public verifiers keys $\{\mathtt{vpk}_j\}_{j\in\mathcal{D}}$; a message μ; and a signature σ, outputs a boolean decision d: $d = 1$ (accept) or $d = 0$ (reject). This algorithm is deterministic.

$\mathsf{Sim}(\mathtt{spk}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}}, \{\mathtt{vsk}_j\}_{j\in\mathcal{C}}, \mu) \to \sigma'$**:** On input public parameters $\mathtt{pp}$, the public signing key $\mathtt{spk}$ of a signer, a set of public signing keys $\{\mathtt{spk}_i\}_{i\in\mathcal{R}}$ belong to a ring of signers (where $\mathtt{spk} \in \{\mathtt{spk}_i\}_{i\in\mathcal{R}}$), the public keys of the designated verifiers $\{\mathtt{vpk}_j\}_{j\in\mathcal{D}}$, the secret keys of the corrupt designated verifiers $\{\mathtt{vsk}_j\}_{j\in\mathcal{C}}$, and a message μ, outputs a simulated signature σ'.

Remark 1. In scenarios with a trusted authority, a master secret key $\mathtt{msk}$ could be produced by Setup, and used for SignKeyGen and VerKeyGen. We do not use such an $\mathtt{msk}$.

Remark 2. We choose to explicitly include the simulation algorithm in the syntax because the off-the-record property demands that it be believable that a given signature was simulated by a corrupt subset of designated verifiers, and for that, the explicit description of a simulation algorithm (as opposed to assurance of its existence) is more convincing.

3.2 Security Model

Following the blueprint of Damgård *et al.* [DHM+20], we describe the intuition behind the security notions formalized in our model.

Correctness, consistency, unforgeability, and off-the-record are obvious translations of existing definitions to our setting. The novelty when modelling security for MDVRS is the introduction of Anonymity.

Correctness: All verifiers $j \in \mathcal{D}$ are able to individually verify an honestly generated signature σ. (This property is exactly the same as for MDVS.)

Consistency: If at least one verifier $j \in \mathcal{D}$ accepts the signature σ, then so do all designated verifiers in $\mathcal{D}$. (This property is the same as for MDVS, and it is not relevant in the single designated verifier setting.)

Unforgeability: Without the knowledge of at least one signer's secret key $\mathtt{ssk}_i$ for a signer in $\mathcal{R}$, no PPT adversary can create a signature σ' for any new message μ' and ring $\mathcal{R}' \subseteq \mathcal{R}$ that is accepted by any set of honest designated verifiers.

Off-The-Record: Given a signature σ, any malicious subset of the designated verifiers $\mathcal{C} \subseteq \mathcal{D}$ cannot convince any other party that σ is a signature from signer $i \in \mathcal{R}$.

Anonymity: Given a signature σ on behalf of a ring $\mathcal{R}$ of (at least two) signers, no PPT adversary can identify which $\mathtt{ssk}_k$ was used to generate σ, even given the knowledge of all designated verifiers' secret keys $\mathtt{vsk}_j$ with $j \in \mathcal{D}$, as well as of all the secret keys belonging to members of $\mathcal{R}$.

Remark 3. When modelling security we explicitly require keys be generated through a key generation oracle, thus excluding rogue key attacks (in which the adversary may not know the secret key corresponding to public key it registers). This is to avoid a conflict that otherwise rises between the notions of unforgeability and off-the-record (simulatability). In detail, to model the off-the-record property we require that an adversary cannot distinguish an honest signature from one simulated by a subset of the designated verifiers. In order to be able to simulate a signature, the subset of designated verifiers' secret key is needed,

otherwise the same simulation could be produced with only public input and without the collaboration of said verifiers, while remaining convincing for any designated verifier (Verify outputs 1), which contradicts unforgeability. Since the verifier secrets are necessary for simulation, the reduction must know the secret key corresponding to any registered public key, which is not guaranteed in case of rogue key attacks.

Next we present the formal definitions of the security notions outlined above. For all future security definitions, we let $\kappa \in \mathbb{N}$ be the security parameter and $u \in \mathbb{N}$ be a fixed upper bound on the number of registered users. For ease of notation we let our adversary be implicitly stateful where ever it is invoked multiple times. This may equivalently be expressed by defining the adversary as a tuple of PPT Turing machines $\mathcal{A} = (\mathcal{A}_1, \ldots, \mathcal{A}_\ell)$, where each $\mathcal{A}_i$ has an additional output state_i which is given as an input to $\mathcal{A}_{i+1}$. Finally, since the oracles and the security experiments share large similarities with the ones for PSDVRS, we highlight the details that differ for one of the schemes in light grey for MDVRS and dark grey for PSDVRS.

All security games share a common setup, which generates public parameters and initialises the necessary oracles. We briefly describe our oracles, referring to Figure 2 for a formal description.

- The key generation oracle $O\mathsf{KG}$ allows the adversary to generate keys for the honest signers and verifiers, receiving their public key and to register the keys of corrupt parties by providing their key generation randomness (to rule out rogue key attacks).
- The corruption oracle $O\mathsf{Corr}$ allows the adversary to corrupt a previously honest party, obtaining their secret key.
- The signing and verification oracles $O\mathsf{Sign}$, $O\mathsf{Verify}$ allow the adversary to respectively obtain signatures for honest signers and verify signatures for honest designated verifiers.

Common setup

$1 : \mathsf{L_S} \leftarrow \varnothing, \mathsf{L_V} \leftarrow \varnothing, \mathsf{L_{corr}} \leftarrow \varnothing, \mathsf{L}_\sigma \leftarrow \varnothing, \mathsf{L_{ver}} \leftarrow \varnothing$

$2 : \mathsf{pp} \leftarrow \mathsf{Setup}(1^\kappa)$

$3 : O \leftarrow \{O\mathsf{KG}, O\mathsf{Sign}, O\mathsf{Verify}, O\mathsf{Corr}\}$

Figure 1: Setup for each of the security games. The oracles are described in Fig. 2.

To allow bookkeeping among the oracles several lists are maintained. For key management this includes a list of signer keys $\mathsf{L_S}$, a list of verifier keys $\mathsf{L_V}$, and a list of corrupt identities $\mathsf{L_{corr}}$. For queries, a list of signing queries L_σ and verification queries $\mathsf{L_V}$ are additionally maintained.

Remark 4. Our security model implicitly assumes that adaptive corruptions only allow the adversary to obtain the long term secrets of honest parties. Looking ahead, this saves the simulator from providing random coins explaining signatures and other values produced by the oracles for previously honest parties. This setting can be achieved by assuming reliable erasures.

We begin by defining correctness for an MDVRS scheme.

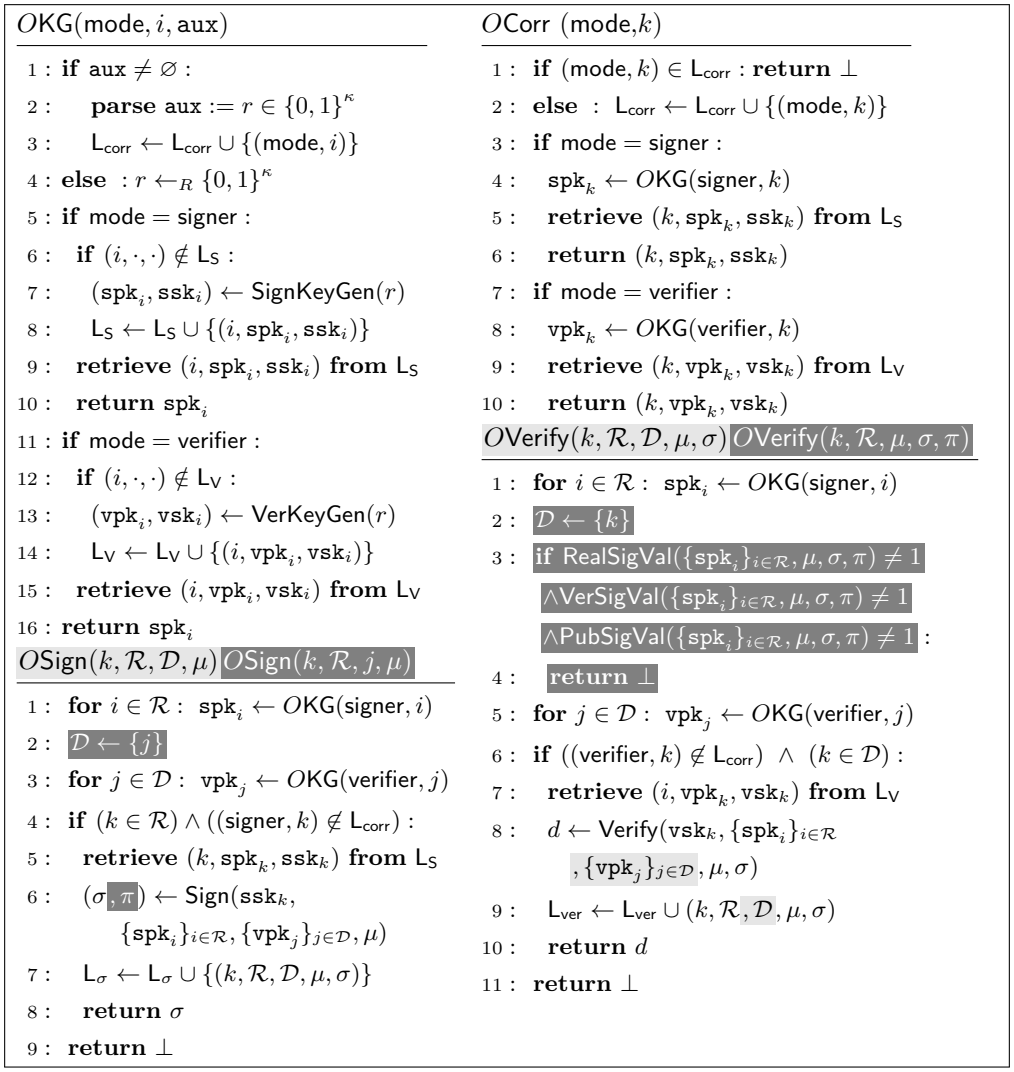

Figure 2: Oracles that an adversary $\mathcal{A}$ in the security games. $\mathsf{L}_{\mathsf{sign}}$ is the list of messages queried to the signing oracle. Details which only feature in the oracles for one of the schemes are marked in light grey for MDVRS and dark grey for PSDVRS.

Definition 2 (Correctness)**.** Let MDVRS be a multi designated verifier ring signature scheme. We say that MDVRS is *correct* if for all PPT adversaries $\mathcal{A}$ taking part in $\mathsf{Experiment}^{\mathrm{COR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$ defined in Figure 3, it holds that

$$\mathsf{Adv}^{\mathrm{COR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\mathrm{COR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)\right] \leq \mathtt{negl}(\kappa).$$

$\mathsf{Experiment}^{\mathrm{COR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$ / $\mathsf{Experiment}^{\mathrm{COR}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$

1 : $(i^*, \mathcal{R}^*, j^*, \mathcal{D}^*, \mu^*) \leftarrow \mathcal{A}^O(\mathtt{pp})$
2 : **if** $(\mathsf{signer}, i^*) \notin \mathsf{L}_{\mathsf{corr}}$:
3 : $\quad (\sigma, \pi) \leftarrow \mathsf{Sign}(\mathtt{ssk}_{i^*}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}^*}, \mathtt{vpk}_{j^*}, \mu^*)$
4 : $b \leftarrow \mathsf{Verify}(\mathtt{vsk}_{j^*}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}^*}, \mu^*, \sigma)$
5 : **if** $(i^* \notin \mathcal{R}^*) \vee (j^* \notin \mathcal{D}^*)$: **return** $\mathtt{lose}$
6 : **if** $(b \neq 1)$:
7 : $\quad$ **return** $\mathtt{win}$
8 : **return** $\mathtt{lose}$

Figure 3: The correctness experiment for MDVRS (Definition 2) and PSDVRS (Definition 8)

As was the case for MDVS we have more than one designated verifier, so it is important to enforce consistency among the verification outcome of independent verifying parties. This is the aim of the following definition.

Definition 3 (Consistency)**.** Let MDVRS be a multi designated verifier ring signature scheme. We say that MDVRS is *consistent* if for all PPT adversaries $\mathcal{A}$ taking part in $\mathsf{Experiment}^{\mathrm{CON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$ (Figure 4), it holds that

$$\mathsf{Adv}^{\mathrm{CON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\mathrm{CON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)\right] \leq \mathtt{negl}(\kappa).$$

$\mathsf{Experiment}^{\mathrm{CON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$

1 : $(\mathcal{R}^*, \mathcal{D}^*, \mu^*, \sigma^*) \leftarrow \mathcal{A}^O(\mathtt{pp})$
2 : // correctness ensures consistency for honestly produced signatures
3 : **if** $(\cdot, \mathcal{R}^*, \mathcal{D}^*, \mu^*, \sigma^*) \in \mathsf{L}_\sigma$: **return** $\mathtt{lose}$
4 : // check consistency of any two verifiers outputs
5 : **for** $j_0 \in \mathcal{D}^*, j_1 \in \mathcal{D}^*, j_0 \neq j_1$:
6 : $\quad b_0 \leftarrow \mathsf{Verify}(\mathtt{vsk}_{j_0}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}^*}, \mu^*, \sigma^*)$
7 : $\quad b_1 \leftarrow \mathsf{Verify}(\mathtt{vsk}_{j_1}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}^*}, \mu^*, \sigma^*)$
8 : $\quad$ **if** $(b_0 \neq b_1) \wedge ((\mathsf{verifier}, j_0) \notin \mathsf{L}_{\mathsf{corr}}) \wedge ((\mathsf{verifier}, j_1) \notin \mathsf{L}_{\mathsf{corr}})$:
9 : $\quad\quad$ **return** $\mathtt{win}$
10 : **return** $\mathtt{lose}$

Figure 4: Consistency experiment for MDVRS (Definition 3)

Notably, the winning conditions in the consistency experiment do *not* check whether the signers' secret keys are corrupted or not. This is because consistency should indeed hold against any PPT adversary as long as not all of the keys of the designated verifiers are corrupted. In particular, the knowledge of a secret signing key does not give any advantage

in creating an inconsistent signature that will be accepted by some designated verifiers and rejected by others.

By design, corrupt designated verifiers can construct an inconsistent signature, since some verifiers will accept it (i.e. those verifiers that created it), while the remaining honest designated verifiers will reject the simulated signature.

In the consistency game we disregard any signatures produced by the signing oracle, as these are already guaranteed to verify by the correctness property.

Definition 4 (Unforgeability)**.** Let MDVRS be a multi designated verifier ring signature scheme. We say that MDVRS is *unforgeable* if for all PPT adversaries $\mathcal{A}$ taking part in $\mathsf{Experiment}^{\mathrm{EUF}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$ (Figure 5), it holds that

$$\mathsf{Adv}^{\mathrm{EUF}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\mathrm{EUF}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)\right] \leq \frac{1}{2} + \mathtt{negl}(\kappa).$$

$\mathsf{Experiment}^{\mathrm{EUF}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$ | $\mathsf{Experiment}^{\mathrm{EUF}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$

1 : $(\mathcal{R}^*, j^*, \mathcal{D}^*, \mu^*, \sigma^*) \leftarrow \mathcal{A}^{O}(\mathtt{pp})$
2 : **if** $(\exists i \in \mathcal{R}^* : (\mathsf{signer}, i) \in \mathsf{L}_{\mathsf{corr}})$: **return** `lose`
3 : **if** $((\mathsf{verifier}, j^*) \in \mathsf{L}_{\mathsf{corr}}) \vee (j^* \notin \mathcal{D}^*)$: **return** `lose`
4 : **if** $(\cdot, \mathcal{R}^*, \mathcal{D}^*, j^*, \mu^*, \cdot) \in \mathsf{L}_\sigma$: **return** `lose`
5 : **if** $(\mathsf{Verify}(\mathtt{vsk}_{j^*}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}^*}, \mu^*, \sigma^*) = 1)$: **return** `win`
6 : **else** : **return** `lose`

Figure 5: The unforgeability experiment for MDVRS (Definition 4) and PSDVRS (Definition 9).

We consider a signature on a message to be a forgery if all signers in the ring are honest and the signature verifies for an honest designated verifier, while no query has been made to the signing oracle for the exact same ring and set of designated verifiers. This definition of unforgeability is quite strong and could potentially be relaxed to allow subsets of designated verifiers or supersets of the ring.

Definition 5 (Off-The-Record)**.** Let MDVRS be a multi designated verifier ring signature scheme. We say that MDVRS is *off-the-record* if for all PPT adversaries $\mathcal{A}$ taking part in $\mathsf{Experiment}^{\mathrm{OTR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$ (Figure 6), it holds that

$$\mathsf{Adv}^{\mathrm{OTR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\mathrm{OTR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)\right] - \frac{1}{2} \leq \mathtt{negl}(\kappa).$$

Definition 5 states that any adversary that corrupts a subset (of size t) of the designated verifiers $\mathcal{C}^*$ cannot determine whether the received signature was created by real signer i^* or simulated by the corrupt verifiers $\mathcal{C}^*$. The adversary is not allowed to see the secret keys for the designated verifiers that are in $\mathcal{D}^* \backslash \mathcal{C}^*$. If the adversary was allowed to get secret keys of additional parties in $\mathcal{D}^*$ (which are not in $\mathcal{C}^*$), then he would be able to distinguish trivially, since any honest designated verifiers (i.e. any $j \in \mathcal{D}^* \backslash \mathcal{C}^*$) can distinguish simulated signatures from real signatures (from the unforgeability property).

Note that Figure 6 prevents the adversary from interacting with the verification oracle using the challenge $(\mathcal{R}^*, \mathcal{D}^*, \mu^*)$ as part of the input. Without this constraint, $\mathcal{A}$ may use the verification oracle to trivially distinguish simulated signatures from real ones (since correctness imposes that designated verifiers be able to make this distinction).

Definition 6 (Signer Anonymity)**.** Let MDVRS be a multi designated verifier ring signature scheme. We say that MDVRS is *signer anonymous* if for all PPT adversaries $\mathcal{A}$ taking

$\mathsf{Experiment}^{\mathrm{OTR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$

$(i^*, \mathcal{R}^*, \mathcal{D}^*, \mu^*) \leftarrow \mathcal{A}^O(\mathtt{pp})$
$\mathcal{C}^* \leftarrow \{(\mathsf{verifier}, j)\}_{j \in \mathcal{D}^*} \cap \mathsf{L}_{\mathsf{corr}}$
$b \leftarrow_R \{0, 1\}$
$\sigma_0 \leftarrow \mathsf{Sign}(\mathtt{ssk}_{i^*}, \{\mathtt{spk}_i\}_{i \in \mathcal{R}^*}, \{\mathtt{vpk}_j\}_{j \in \mathcal{D}^*}, \mu^*)$
$\sigma_1 \leftarrow \mathsf{Sim}(\mathtt{spk}_{i^*}, \{\mathtt{spk}_i\}_{i \in \mathcal{R}^*}, \{\mathtt{vpk}_j\}_{j \in \mathcal{D}^*}, \{\mathtt{vsk}_j\}_{(\mathsf{verifier}, j) \in \mathcal{C}^*}, \mu^*)$
$b' \leftarrow \mathcal{A}^O(\sigma_b)$
if $C^* \neq \{(\mathsf{verifier}, j)\}_{j \in \mathcal{D}^*} \cap \mathsf{L}_{\mathsf{corr}}$: **return** `lose`
if $(\mathsf{signer}, i^*) \in \mathsf{L}_{\mathsf{corr}}$: **return** `lose`
if $(\cdot, \mathcal{R}^*, \mathcal{D}^*, \mu^*, \cdot) \in \mathsf{L}_{\mathsf{ver}}$: **return** `lose`
if $b' = b$: **return** `win`
return `lose`

Figure 6: The off-the-record experiment for MDVRS (Definition 5).

part in the anonymity experiment $\mathsf{Experiment}^{\mathrm{ANON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$ (Figure 7), it holds that

$$\mathsf{Adv}^{\mathrm{ANON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\mathrm{ANON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)\right] - \frac{1}{2} \leq \mathtt{negl}(\kappa).$$

To satisfy anonymity we require that any signature produced by an honest signer should be indistinguishable from a signature produced by a different honest signer in the same ring. We require this to hold even against full key exposure: indeed there is no restriction on the adversary's interaction with the corruption oracle.

$\mathsf{Experiment}^{\mathrm{ANON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$ $\mathsf{Experiment}^{\mathrm{ANON}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$

$(i_0^*, i_1^*, \mathcal{R}^*, \mathcal{D}^*, j^*, \mu^*) \leftarrow \mathcal{A}^O(\mathtt{pp})$
$b \leftarrow_R \{0, 1\}$
$(\sigma_b, \pi) \leftarrow \mathsf{Sign}(\mathtt{ssk}_{i_b^*}, \{\mathtt{spk}_i\}_{i \in \mathcal{R}^*}, \{\mathtt{vpk}_j\}_{j \in \mathcal{D}^*}, \mathtt{vpk}_{j^*}, \mu^*)$
$b' \leftarrow \mathcal{A}^O(\sigma_b)$
if $(i_0^* \notin \mathcal{R}^*) \vee (i_1^* \notin \mathcal{R}^*)$: **return** `lose`
if $b' = b$: **return** `win`
else : **return** `lose`

Figure 7: The anonymity experiment for MDVRS (Definition 6) and PSDVRS (Definition 10).

Definition 7. A secure MDVRS scheme $\mathsf{MDVRS} = (\mathsf{Setup}, \mathsf{SignKeyGen}, \mathsf{VerKeyGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{Sim})$ must have the properties correctness (Definition 2), consistency (Definition 3), unforgeability (Definition 4) and off-the-record (Definition 5).

4 Provably Simulatable Designated Verifier Ring Signatures

Rather than constructing MDVRS directly we follow the approach of Damgård *et al.* [DHM+20] to break down the problem. This approach consists of two key elements: producing designated verifier ring signatures (DVRS) for each designated verifier, and

proving that these signatures were produced correctly. The naïve approach for doing this becomes problematic when considering the off-the-record property: each designated verifiers j would not be able to simulate a signature for other designated verifiers (unless $j \in \mathcal{D}$). To resolve this, the proof must show that *either* each of the DVRS signatures verifies, *or* they are all simulated.

Consider a DVRS scheme. Such a scheme consists of five algorithms: Setup, SignKeyGen, VerKeyGen, Sign and Verify. To satisfy the off-the-record property there must additionally be an efficient simulation algorithm Sim, allowing a verifier to produce a signature indistinguishable from a real signature. While the Sim algorithm is never meant to be used in practice, its existence is essential in guaranteeing the properties of DVRS and its description plays a central role in the security proofs. In order to leverage DVRS in the building of MDVRS, we need another simulation algorithm: one that can be run publicly to produce a signature that looks real (without knowledge of the verifier's secret key), but does not verify (given that knowledge). Consistency for our MDVRS requires that it be possible to prove how each individual signature was produced.

We introduce an intermediate primitive, Provably Simulatable DVRS (PSDVRS), which supports exactly this. A PSDVRS scheme is like a DVRS with two distinct simulation algorithms: PubSigSim and VerSigSim, for public and verifier simulation respectively. It also has three algorithms RealSigVal, PubSigVal and VerSigVal which validate how a signature or simulation was produced.

4.1 Syntax

We present the algorithms in the PSDVRS syntax as needed, starting with the core algorithms, and introducing the simulation and validation algorithms as required by the security properties.

The algorithms Setup, SignKeyGen and VerKeyGen of a PSDVRS scheme have the same syntax as in an MDVRS scheme. Recall all algorithms take the public parameters produced by Setup as an implicit argument.The signing and verification algorithms of PSDVRS differ from the ones of MDVRS in that the signatures only target a single verifier.

Setup, SignKeyGen, VerKeyGen are as in Definition 1.

Sign$(\mathtt{ssk}_k, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}, \mu) \to (\sigma, \pi)$ On input public parameters pp;a signer secret key $\mathtt{ssk}_k$; a ring of signer public keys $\{\mathtt{spk}_i\}_{i\in\mathcal{R}}$ (subject to $k \in \mathcal{R}$, enforcing the inclusion of $\mathtt{spk}_k$ in the ring); a designated verifier public key vpk; and message μ; produces a signature σ along with a proof π that the signature was produced by an honest signer.

Verify$(\mathtt{vsk}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mu, \sigma) = d \in \{0, 1\}$ On input public parameters pp;a verifier secret key vsk; a ring of signer keys $\{\mathtt{spk}_i\}_{i\in\mathcal{R}}$; a message μ; and a signature σ outputs a boolean decision d: $d = 1$ (accept) or $d = 0$ (reject). This algorithm is deterministic.

4.2 PSDVRS Security Model

The first properties which a PSDVRS scheme must statisfy are the standard notions of correctness and existential unforgeability, which we describe formally below. The oracles used in the experiments described in this section are collected in Fig. 2.

Correctness: An honestly generated signature should verify for the designated verifier.

Definition 8 (Correctness)**.** Let PSDVRS be a provably simulatable designated verifier ring signature scheme. We say that PSDVRS is *correct* if for all PPT adversaries $\mathcal{A}$ taking

part in the correctness experiment $\mathsf{Experiment}^{\text{COR}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ defined in Figure 3, it holds that

$$\mathsf{Adv}^{\text{COR}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\text{COR}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)\right] \leq \mathtt{negl}(\kappa).$$

Existential Unforgeability: Without knowledge of the verifier's secret key or at least one signer secret key for a given ring $\mathcal{R}$, no PPT adversary can produce a verifying signature on a message μ (and ring $\mathcal{R}$) for which it has not previously seen a signature.

Definition 9 (Existential Unforgeability)**.** Let PSDVRS be a provably simulatable designated-verifier ring signature scheme. We say that PSDVRS has *Existential Unforgeability* if for all PPT adversaries $\mathcal{A}$ taking part in the anonymity experiment $\mathsf{Experiment}^{\text{EUF}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ (Figure 5), it holds that

$$\mathsf{Adv}^{\text{EUF}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\text{EUF}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)\right] \leq \mathtt{negl}(\kappa).$$

4.2.1 Signer Anonymity

To achieve signer anonymity in and MDVRS scheme the PSDVRS scheme we construct it from must also have signer anonymity, that is, a verifier should be unable to distinguish which honest member of a ring produced a signature. On a high level, no PPT adversary can distinguish signatures produced by two honest signers $i_0, i_1 \in \mathcal{R}$ for the same message μ and designated verifier j, even when all secret keys are given to the adversary; we formalise anonymity in the following definition.

Definition 10 (Signer Anonymity)**.** Let PSDVRS be a provably simulatable designated-verifier ring signature scheme. We say that PSDVRS is *signer anonymous* if for all PPT adversaries $\mathcal{A}$ taking part in the anonymity experiment $\mathsf{Experiment}^{\text{PSDVRS-ANON}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ (Figure 7), it holds that

$$\mathsf{Adv}^{\text{PSDVRS-ANON}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\text{PSDVRS-ANON}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)\right] - \frac{1}{2} \leq \mathtt{negl}(\kappa).$$

4.2.2 Provable Signing

A signer must be able to prove a signature was is real, to accommodate this the output of the Sign algorithm must be extended to include a proof π. This proof may be verified by a final algorithm:

$\mathsf{RealSigVal}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}, \mu, \sigma, \pi) \rightarrow d \in \{0, 1\}$ On input public parameters pp; a ring of signers $\{\mathtt{spk}_i\}_{i\in\mathcal{R}}$; a verifier public key vpk; a message μ; a signature σ; and proof π outputs a boolean decision d: $d = 1$ (accept) or $d = 0$ (reject).

A real signature along with its corresponding proof should always cause RealSigVal to accept. Additionally, it should be intractable to produce a signature and proof such that RealSigVal outputs accept, but the signature does not verify.

Definition 11 (Provable Signing Correctness)**.** Let PSDVRS be a provably simulatable designated verifier ring signature scheme. We say that PSDVRS has *Provable Signing Correctness* if for all PPT adversaries $\mathcal{A}$ taking part in $\mathsf{Experiment}^{\text{ProvSig-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ (Figure 8), it holds that

$$\mathsf{Adv}^{\text{ProvSig-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\text{ProvSig-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)\right] \leq \mathtt{negl}(\kappa).$$

$\mathsf{Experiment}^{\text{ProvSig-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$

1 : $(i^*, \mathcal{R}^*, j^*, \mu^*) \leftarrow \mathcal{A}^{O}(\mathtt{pp})$
2 : **if** $(\mathsf{signer}, i^*) \in \mathsf{L}_{\mathsf{corr}}$: **return** `lose`
3 : **if** $i^* \notin \mathcal{R}^*$: **return** `lose`
4 : $(\sigma, \pi) \leftarrow \mathsf{Sign}(\mathtt{ssk}_{i^*}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \mathtt{vpk}_{j^*}, \mu^*)$
5 : **if** $\mathsf{RealSigVal}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \mathtt{vpk}_{j^*}, \mu^*, \sigma, \pi) \neq 1$: **return** `win`
6 : **return** `lose`

Figure 8: The provable signing correctness experiment for PSDVRS (Definition 11).

$\mathsf{Experiment}^{\text{ProvSig-Sound}}_{\mathsf{PSDVRS},n,\mathcal{A}}(\kappa, u)$

1 : $(\mathcal{R}^*, j^*, \mu^*, \sigma^*, \pi^*) \leftarrow \mathcal{A}^{O}(\mathtt{pp})$
2 : **if** $\big(\mathsf{Verify}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \mathtt{vsk}_{j^*}, \mu^*, \sigma^*) = 0 \ \wedge$
$\mathsf{RealSigVal}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \mathtt{vpk}_{j^*}, \mu^*, \sigma^*, \pi^*) = 1\big)$: **return** `win`
3 : **else** : **return** `lose`

Figure 9: The provable signing soundness experiment for PSDVRS (Definition 12).

Definition 12 (Provable Signing Soundness)**.** Let PSDVRS be a provably simulatable designated verifier ring signature scheme. We say that PSDVRS has *Provable Signing Soundness* if for all n and all PPT adversaries $\mathcal{A}$ taking part in the experiment $\mathsf{Experiment}^{\text{ProvSig-Sound}}_{\mathsf{PSDVRS},n,\mathcal{A}}(\kappa, u)$ (Figure 9), it holds that

$$\mathsf{Adv}^{\text{ProvSig-Sound}}_{\mathsf{PSDVRS},n,\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\text{VerSigSim-Sound}}_{\mathsf{PSDVRS},n,\mathcal{A}}(\kappa, u)\right] \leq \mathtt{negl}(\kappa).$$

4.2.3 Provable Public Simulation

As any subset of verifiers should be able to simulate an MDVRS signature, we require a path for simulating on behalf of any non-colluding verifiers. This approach for simulating should also be provable, to allow a consistency proof in an MDVRS construction. To allow provable public simulation the syntax must be extended with the following algorithms:

$\mathsf{PubSigSim}(\mathtt{spk}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}, \mu) \rightarrow (\sigma, \pi)$ On input public parameters `pp`; a signer public key `spk`; a ring of signers $\{\mathtt{spk}_i\}_{i\in\mathcal{R}}$ (where $\mathtt{spk} \in \{\mathtt{spk}_i\}_{i\in\mathcal{R}}$); a verifier public key `vpk`; and message μ outputs a simulated signature σ and a proof π that σ is a public simulation.

$\mathsf{PubSigVal}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}, \mu, \sigma, \pi) \rightarrow d \in \{0, 1\}$ On input public parameters `pp`; a ring of signers $\{\mathtt{spk}_i\}_{i\in\mathcal{R}}$; a verifier public key `vpk`; a message μ; a signature σ; and proof π outputs a boolean decision d: $d = 1$ (accept) or $d = 0$ (reject).

These algorithms allow producing public simulations which other outsiders cannot distinguish from real signatures, while being able to prove a signature was publicly simulated. Additionally, an adversary should not be able to produce a signature σ which is accepted by Verify, along with a proof causing σ to be accepted as a public simulation by PubSigVal. These notions are expressed formally in Definiton 13, 14 and 15.

$\mathsf{Experiment}^{\text{PubSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$

1 : $(i^*, \mathcal{R}^*, j^*, \mu^*) \leftarrow \mathcal{A}^O(\mathtt{pp})$
2 : $b \leftarrow_R \{0,1\}$
3 : **if** $(\mathsf{signer}, i^*) \in \mathsf{L}_{\mathsf{corr}} \vee (\mathsf{signer}, i^*) \notin \mathsf{L}_{\mathsf{S}}$: **return** `lose`
4 : $(\sigma_0, \pi_0) \leftarrow \mathsf{Sign}(\mathtt{ssk}_{i^*}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \mathtt{vpk}_{j^*}, \mu^*)$
5 : $(\sigma_1, \pi_1) \leftarrow \mathsf{PubSigSim}(\mathtt{spk}_{i^*}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \mathtt{vpk}_{j^*}, \mu^*)$
6 : $b' \leftarrow \mathcal{A}^O(\sigma_b)$ // The adversary may now corrupt more identities
7 : **if** $(\mathsf{signer}, i^*) \in \mathsf{L}_{\mathsf{corr}}$: **return** `lose`
8 : **if** $i^* \notin \mathcal{R}^*$: **return** `lose`
9 : **if** $((\mathsf{verifier}, j^*) \in \mathsf{L}_{\mathsf{corr}})$: **return** `lose`
10 : **if** $(j^*, \mathcal{R}^*, \mu^*, \cdot) \in \mathsf{L}_{\mathsf{ver}}$: **return** `lose`
11 : **if** $b' = b$: **return** `win`
12 : **else** : **return** `lose`

Figure 10: The public signature simulation indistinguishability experiment for PSDVRS (Definition 13). Note, the adversary is never given the proof π_b as this proof is only used in the witness for a larger proof when constructing MDVRS. In fact, given this proof the adversary could trivially distinguish the real and simulated case.

Definition 13 (PubSigSim Indistinguishability)**.** Let PSDVRS be a provably simulatable designated verifier ring signature scheme. We say that PSDVRS has *PubSigSim Indistinguishability* if for all PPT adversaries $\mathcal{A}$ taking part in the experiment $\mathsf{Experiment}^{\text{PubSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ (Figure 10), it holds that

$$\mathsf{Adv}^{\text{PubSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\text{PubSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)\right] - \frac{1}{2} \leq \mathtt{negl}(\kappa).$$

Definition 14 (PubSigSim Correctness)**.** Let PSDVRS be a provably simulatable designated-verifier ring signature scheme. We say that PSDVRS has *PubSigSim Correctness* if for all PPT adversaries $\mathcal{A}$ taking part in $\mathsf{Experiment}^{\text{PubSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ (Figure 11), it holds that

$$\mathsf{Adv}^{\text{PubSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\text{PubSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)\right] \leq \mathtt{negl}(\kappa).$$

$\mathsf{Experiment}^{\text{PubSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$

1 : $(i^*, \mathcal{R}^*, j^*, \mu^*) \leftarrow \mathcal{A}^O(\mathtt{pp})$
2 : $(\sigma, \pi) \leftarrow \mathsf{PubSigSim}(\mathtt{spk}_{i^*}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \mathtt{vpk}_{j^*}, \mu^*)$
3 : **if** $i^* \notin \mathcal{R}^*$: **return** `lose`
4 : **if** $\mathsf{PubSigVal}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \mathtt{vpk}_{j^*}, \mu^*, \sigma, \pi) \neq 1$: **return** `win`
5 : **return** `lose`

Figure 11: The public signature simulation correctness experiment for PSDVRS (Definition 14).

Definition 15 (PubSigSim Soundness)**.** Let PSDVRS be a provably simulatable designated-verifier ring signature scheme. We say that PSDVRS has *PubSigSim Soundness* if for all PPT adversaries $\mathcal{A}$ taking part in the experiment $\mathsf{Experiment}^{\text{PubSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ (Figure 12), it holds that

$$\mathsf{Adv}^{\text{PubSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\text{PubSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)\right] \leq \mathtt{negl}(\kappa).$$

$\mathsf{Experiment}^{\textsc{PubSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$

$(\mathcal{R}^*, j^*, \mu^*, \sigma^*, \pi^*) \leftarrow \mathcal{A}^{O}(\mathtt{pp})$

if $((\mathsf{verifier}, j^*) \in \mathsf{L}_{\mathsf{corr}}) \vee ((\mathsf{verifier}, j^*) \notin \mathsf{L}_{\mathsf{V}})$: **return** `lose`

if $\big(\mathsf{Verify}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \mathtt{vsk}_{j^*}, \mu^*, \sigma^*) = 1) \wedge$
$(\mathsf{PubSigVal}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \mathtt{vpk}_{j^*}, \mu^*, \sigma^*, \pi^*) = 1\big)$: **return** `win`

else : **return** `lose`

Figure 12: The public signature simulation soundness experiment for PSDVRS (Definition 15).

4.2.4 Provable Verifier Simulation

To construct MDVRS colluding verifiers should also be able to produce simulations on their own behalf, which remain indistinguishable from real signatures even if a verifier gives away their secret to a third party. We once again expand the syntax with two additional algorithms, to allow provable verifier simulation:

$\mathsf{VerSigSim}(\mathtt{spk}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}, \mathtt{vsk}, \mu) \rightarrow (\sigma, \pi)$ On input public parameters `pp`; a signer public key `spk`; a ring of signers $\{\mathtt{spk}_i\}_{i\in\mathcal{R}}$ (where $\mathtt{spk} \in \{\mathtt{spk}_i\}_{i\in\mathcal{R}}$); a verifier secret key `vsk`; and message μ outputs a simulated signature σ and a proof π that σ is a public simulation.

$\mathsf{VerSigVal}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}, \mu, \sigma, \pi) \rightarrow d \in \{0, 1\}$ On input public parameters `pp`; a ring of signers $\{\mathtt{spk}_i\}_{i\in\mathcal{R}}$; a verifier public key `vpk`; a message μ; a signature σ; and proof π outputs a boolean decision d: $d = 1$ (accept) or $d = 0$ (reject).

Verifier simulations must be indistinguishable from real signatures, even given the secret key of the verifier. Furthermore, the proof π and simulated signature σ should be accepted as a verifier simulation by $\mathsf{VerSigVal}$, when corretly produced, while it remains hard to satisfy $\mathsf{VerSigVal}$ without the verifier secret key. This is formalised in Definition 16, 17 and 18.

Definition 16 (VerSigSim Indistinguishability)**.** Let PSDVRS be a provably simulatable designated verifier ring signature scheme. We say that PSDVRS has *VerSigSim Indistinguishability* if for all PPT adversaries $\mathcal{A}$ taking part in the experiment $\mathsf{Experiment}^{\textsc{VerSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ (Figure 13), it holds that

$$\mathsf{Adv}^{\textsc{VerSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\textsc{VerSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)\right] - \frac{1}{2} \leq \mathtt{negl}(\kappa).$$

Again, we remark that the adversary is never given π_b as this can be used to trivially distinguish the real and simulated case. However, the proof is part of the witness for a larger proof when constructing MDVRS.

Definition 17 (VerSigSim Correctness)**.** Let PSDVRS be a provably simulatable designated-verifier ring signature scheme. We say that PSDVRS has *VerSigSim Correctness* if for all PPT adversaries $\mathcal{A}$ taking part in $\mathsf{Experiment}^{\textsc{VerSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ (Figure 14), it holds that

$$\mathsf{Adv}^{\textsc{VerSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\textsc{VerSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)\right] \leq \mathtt{negl}(\kappa).$$

Definition 18 (VerSigSim Soundness)**.** Let PSDVRS be a provably simulatable designated-verifier ring signature scheme. We say that PSDVRS has *VerSigSim Soundness* if for all n and all PPT adversaries $\mathcal{A}$ taking part in $\mathsf{Experiment}^{\textsc{VerSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ (Figure 15), it holds that

$$\mathsf{Adv}^{\textsc{VerSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = \Pr\left[\mathtt{win} \leftarrow \mathsf{Experiment}^{\textsc{VerSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)\right] \leq \mathtt{negl}(\kappa).$$

$\mathsf{Experiment}^{\text{VerSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$

1 : $(i^*, \mathcal{R}^*, j^*, \mu^*) \leftarrow \mathcal{A}^O(\mathtt{pp})$
2 : **if** $((\mathsf{signer}, i^*) \in \mathsf{L}_{\mathsf{corr}})$: **return** `lose` // ensure the challenger is able to sign
3 : $b \leftarrow_R \{0,1\}$
4 : $(\sigma_0, \pi_0) \leftarrow \mathsf{Sign}(\mathtt{ssk}_{i^*}, \{\mathtt{spk}_i\}_{i \in \mathcal{R}^*}, \mathtt{vpk}_{j^*}, \mu^*)$
5 : $(\sigma_1, \pi_1) \leftarrow \mathsf{VerSigSim}(\mathtt{spk}_{i^*}, \{\mathtt{spk}_i\}_{i \in \mathcal{R}^*}, \mathtt{vsk}_{j^*}, \mu^*)$
6 : $b' \leftarrow \mathcal{A}^O(\sigma_b)$
7 : **if** $((\mathsf{signer}, i^*) \in \mathsf{L}_{\mathsf{corr}}) \vee (i^* \notin \mathcal{R}^*)$: **return** `lose`
8 : **if** $(j^*, \mathcal{R}^*, \mu^*, \cdot) \in \mathsf{L}_{\mathsf{ver}}$: **return** `lose`
9 : **if** $b' = b$: **return** `win`
10 : **else** : **return** `lose`

Figure 13: The verifier signature simulation indistinguishability experiment for PSDVRS (Definition 16).

$\mathsf{Experiment}^{\text{VerSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$

1 : $(i^*, \mathcal{R}^*, j^*, \mu^*) \leftarrow \mathcal{A}^O(\mathtt{pp})$
2 : $(\sigma, \pi) \leftarrow \mathsf{VerSigSim}(\mathtt{spk}_{i^*}, \{\mathtt{spk}_i\}_{i \in \mathcal{R}^*}, \mathtt{vsk}_{j^*}, \mu^*)$
3 : **if** $i^* \notin \mathcal{R}^*$: **return** `lose`
4 : **if** $\mathsf{VerSigVal}(\{\mathtt{spk}_i\}_{i \in \mathcal{R}^*}, \mathtt{vpk}_{j^*}, \mu^*, \sigma, \pi) \neq 1$: **return** `win`
5 : **else** : **return** `lose`

Figure 14: The verifier signature simulation correctness game for PSDVRS (Definition 17).

$\mathsf{Experiment}^{\text{VerSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$

1 : $(\mathcal{R}^*, j^*, \mu^*, \sigma^*, \pi^*) \leftarrow \mathcal{A}^{O}(\mathtt{pp}, \mathtt{vpk})$

2 : **if** $\big((\mathsf{verifier}, j^*) \notin \mathsf{L}_{\mathsf{corr}}\big) \ \wedge$
$\mathsf{VerSigVal}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}^*}, \mathtt{vpk}, \mu^*, \sigma^*, \pi^*) = 1$: **return** `win`

3 : **else** : **return** `lose`

Figure 15: The verifier signature simulation soundness experiment for PSDVRS (Definition 17).

Definition 19. A secure PSDVRS consisting of the algoriths $\mathsf{Setup}, \mathsf{VerKeyGen}, \mathsf{SignKeyGen}, \mathsf{Sign}, \mathsf{Verify}$, simulation algorithms $\mathsf{PubSigSim}, \mathsf{VerSigSim}$, and validation algorithms $\mathsf{PubSigVal}, \mathsf{VerSigVal}, \mathsf{RealSigVal}$, must satisfy:

- Correctness (Definition 8), Existential Unforgeablility (Definition 9) and Signer Anonymity (Definition 10)
- Provable Signing correctness (Definition 11) and soundness (Definition 12)
- PubSigSim indistinguishability (Definition 13), correctness (Definition 14) and soundness (Definition 15)
- VerSigSim indistinguishability (Definition 16), correctness (Definition 17) and soundness (Definition 17)

5 Constructing MDVRS from PSDVRS

Our goal is to construct an MDVRS scheme given a PSDVRS scheme. To produce an MDVRS signature we may first produce a set of signatures targeted toward each designated verifier. To ensure consistent verification we must prove that all signatures are real signatures for the same message. To support simulation by a subset of the verifiers — which should be indistinguishable even given their secret keys — the relation must additionally admit sets which are a combination of verifier and public simulations.

5.1 Construction

We use relation $\mathscr{R}_{cons}$ (where "cons" stands for "consistency") in our construction, to enforce consistency across Provably Simulatable DVRS. Let $\mathscr{R}_{cons}$ be defined by the following set:

$$\left\{ \begin{array}{c} \phi = \begin{pmatrix} \mathsf{PSDVRS.pp} \\ \{\mathtt{spk}_i\}_{i\in\mathcal{R}} \\ \{\mathtt{vpk}_j\}_{j\in\mathcal{D}} \\ \{\sigma_j\}_{j\in\mathcal{D}} \\ \mu \end{pmatrix} \\ \\ w = \{\pi_j\}_{j\in\mathcal{D}} \end{array} \middle| \begin{array}{c} \Big(\bigwedge_{j\in\mathcal{D}} \mathsf{RealSigVal}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}_j, \mu, \sigma_j, \pi_j)\Big) \\ \\ \bigvee \\ \\ \bigwedge_{j\in\mathcal{D}} \begin{pmatrix} \mathsf{PubSigVal}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}_j, \mu, \sigma_j, \pi_j) \\ \vee \\ \mathsf{VerSigVal}(\{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}_j, \mu, \sigma_j, \pi_j) \end{pmatrix} \end{array} \right\}$$

Informally $\mathscr{R}_{cons}$ ensures that either all signatures are real or all signatures are either a verifier or public simulation.The first branch will be used by an honest signer as it may produce a real PSDVRS targeted toward each verifier. The second branch enables the off-the-record property: each colluding verifier may produce verifier simulation toward themselves, with the remaining signatures being public simulations. In combination, the resulting zero knowledge proof of $\mathscr{R}_{cons}$ and signature simulations are indistinguishable

$\mathsf{Setup}(1^\kappa)$

1 : $\mathtt{pp}_{\mathsf{PSDVRS}} \leftarrow \mathsf{PSDVRS.Setup}(1^\kappa)$
2 : $(\mathtt{crs}, \mathtt{td}) \leftarrow \mathsf{NIZK.Setup}(1^\kappa, \mathscr{R}_{cons})$
3 : **return** $\mathtt{pp} = (\mathtt{pp}_{\mathsf{PSDVRS}}, \mathtt{crs})$

$\mathsf{SignKeyGen}()$

1 : **return** $(\mathtt{spk}, \mathtt{ssk}) \leftarrow \mathsf{PSDVRS.SignKeyGen}()$

$\mathsf{VerKeyGen}()$

1 : **return** $(\mathtt{vpk}, \mathtt{vsk}) \leftarrow \mathsf{PSDVRS.VerKeyGen}()$

$\mathsf{Sign}(\mathtt{ssk}_k, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}}, \mu)$

1 : **for** $j \in \mathcal{D}$: $(\sigma_j, \pi_j) \leftarrow \mathsf{PSDVRS.Sign}(\mathtt{ssk}_k, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}_j, \mu)$
2 : $\phi = (\mathsf{PSDVRS.pp}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}}, \{\sigma_j\}_{j\in\mathcal{D}}, \mu)$
3 : $w = \{\pi_j\}_{j\in\mathcal{D}}$
4 : $\pi \leftarrow \mathsf{NIZK.Prove}(\mathtt{crs}, \phi, w)$
5 : **return** $\sigma = (\{\sigma_j\}_{j\in\mathcal{D}}, \pi)$

$\mathsf{Verify}(\mathtt{vsk}_k, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}}, \mu, \sigma = (\{\sigma_j\}_{j\in\mathcal{D}}, \pi))$

1 : $\phi = (\mathsf{PSDVRS.pp}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}}, \{\sigma_j\}_{j\in\mathcal{D}}, \mu)$
2 : $d_{\mathsf{NIZK}} \leftarrow \mathsf{NIZK.Verify}(\mathtt{crs}, \phi, \pi)$
3 : $d_{sig} \leftarrow \mathsf{PSDVRS.Verify}(\mathtt{vsk}_k, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mu, \sigma_k)$
4 : **return** $d_{\mathsf{NIZK}} \wedge d_{sig}$

$\mathsf{Sim}(\mathtt{spk}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}}, \{\mathtt{vsk}_j\}_{j\in\mathcal{C}}, \mu)$

1 : **for** $j \in \mathcal{C}$: $(\sigma_j, \pi_j) \leftarrow \mathsf{VerSigSim}(\mathtt{spk}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}_j, \mathtt{vsk}_j, \mu)$
2 : **for** $j \in \mathcal{D} \setminus \mathcal{C}$: $(\sigma_j, \pi_j) \leftarrow \mathsf{PubSigSim}(\mathtt{spk}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \mathtt{vpk}_j, \mu)$
3 : $\phi = (\mathsf{PSDVRS.pp}, \{\mathtt{spk}_i\}_{i\in\mathcal{R}}, \{\mathtt{vpk}_j\}_{j\in\mathcal{D}}, \{\sigma_j\}_{j\in\mathcal{D}}, \mu)$
4 : $w = \{\pi_j\}_{j\in\mathcal{D}}$
5 : $\pi \leftarrow \mathsf{NIZK.Prove}(\mathtt{crs}, \phi, w)$
6 : **return** $\sigma = (\{\sigma_j\}_{j\in\mathcal{D}}, \pi)$

Figure 16: Our construction of MDVRS from PSDVRS and NIZK.

from those produced by a real signer, even given the secrets of the colluding verifiers. Allowing this second branch in the consistency proof does not help an adversary forge towards honest designated verifiers as the adversary is unable to produce a proof of provenance that will pass the real signature validation (first branch).

5.2 Security Statement and Proofs

We prove the security of our MDVRS construction by reducing to the security of the underlying primitives. For security the proof simulation and extraction for NIZK must be straight line, i.e. not require rewinding of the adversary during simulation or extraction. See [GKO+23] for an overview of straight line and simulation extractable proof systems.

Theorem 1. *The MDVRS scheme described in Section 5 (Figure 16) is secure according to Definition 7 when instantiated with a secure PSDVRS scheme and a NIZK scheme with perfect completeness, zero knowledge (with straight line simulation and advantage bounded by $\epsilon_{\mathrm{ZK}}^{\mathsf{NIZK}}$) and simulation extractability (with straight line extraction and advantage bounded by $\epsilon_{\text{SIM-EXT}}^{\mathsf{NIZK}}$). Let $\kappa^{\mathsf{NIZK}} = \epsilon_{\mathrm{ZK}}^{\mathsf{NIZK}} + Q_{\mathsf{Verify}} \cdot \epsilon_{\text{SIM-EXT}}^{\mathsf{NIZK}}$, then the MDVRS scheme is secure with*

advantages:

$$\mathsf{Adv}^{\text{COR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa,u) \leq \kappa^{\mathsf{NIZK}} + \epsilon^{\mathsf{PSDVRS}}_{\text{COR}},$$

$$\mathsf{Adv}^{\text{CON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa,u) \leq \kappa^{\mathsf{NIZK}} + \epsilon^{\mathsf{PSDVRS}}_{\text{ProvSig-sound}}$$

$$+\epsilon^{\mathsf{PSDVRS}}_{\text{PubSigSim-sound}} + \epsilon^{\mathsf{PSDVRS}}_{\text{VerSigSim-sound}},$$

$$\mathsf{Adv}^{\text{EUF}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa,u) \leq \kappa^{\mathsf{NIZK}} + \epsilon^{\mathsf{NIZK}}_{\text{SIM-EXT}} + \epsilon^{\mathsf{PSDVRS}}_{\text{EUF}},$$

$$\mathsf{Adv}^{\text{OTR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa,u) \leq \kappa^{\mathsf{NIZK}} + u \cdot (\epsilon^{\mathsf{PSDVRS}}_{\text{VerSigSim-ind}} + \epsilon^{\mathsf{PSDVRS}}_{\text{PubSigSimInd-ind}}),$$

$$\mathsf{Adv}^{\text{ANON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa,u) \leq \kappa^{\mathsf{NIZK}} + u \cdot \epsilon^{\mathsf{PSDVRS}}_{\text{ANON}}.$$

Secure erasure is assumed for the case of adaptive corruptions.

Proof. We prove each of the security properties of the MDVRS scheme described in Figure 16 individually, reducing to the security of the PSDVRS and NIZK primitives.

Remark 5. When reducing to the security of PSDVRS we must provide access to the signing and verification oracle. The signing oracle in the PSDVRS games does not return the proof of how the signature was created, although it outputs proof of provenance π, only the signatures σ are given to the adversary (and hence are available to our reduction). It is therefore necessary to simulate the consistency proofs needed to respond to MDVRS queries during our reductions. Now consider the verification oracle. For our MDVRS construction verification has two parts (1) verifying the zero knowledge proof and (2) verifying a PSDVRS signature. The first of these may be done directly, but verifying a PSDVRS signature requires the verifier secret key or use of the verification oracle in the PSDVRS game we are reducing to. Querying the verification oracle requires the proof of provenance for the PSDVRS signature. This proof of provenance is not provided directly, but may be obtained by extracting from the NIZK.

Correctness. The MDVRS scheme is correct according to Definition 2. First, we define the following sequence of indistinguishable hybrids:

$\mathcal{H}_0$: Run the $\mathsf{Experiment}^{\text{COR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa,u)$ as described in Figure 3.

$\mathcal{H}_1$: Same as $\mathcal{H}_0$, except abort if d_{NIZK} is false during verification.

$\mathcal{H}_2$: Same as $\mathcal{H}_1$, but replace the honestly produced CRS with a simulated one, and simulate all proofs contained in signatures produced by the oracles.

$\mathcal{H}_3$: Same as $\mathcal{H}_2$, but for every verification query extract the witness from the NIZK π (unless the signature was produced by the signing oracle, where the oracle may simply return 1). Abort if extraction fails.

$\mathcal{H}_4$: Same as $\mathcal{H}_3$, except abort if d_{Sig} is false during verification. The adversary can now no longer win, as the signature will always verify.

$\mathcal{H}_0 \approx \mathcal{H}_1$**:** This is indistinguishable by perfect completeness of the NIZK.

$\mathcal{H}_1 \approx \mathcal{H}_2$**:** This is indistinguishable by zero knowledge of the NIZK.

$\mathcal{H}_2 \approx \mathcal{H}_3$**:** This is indistinguishable by the simulation extractability of the NIZK. The adversary can at most query Q_{Verify} signatures.

$\mathcal{H}_3 \approx \mathcal{H}_4$**:** To distinguish these hybrids an adversary must cause d_{sig} to be false, i.e. find inputs to the PSDVRS signing algorithm which cause it to produce signatures which do not verify. Such an MDVRS adversary $\mathcal{A}$ may be reduced to an PSDVRS adversary with the same advantage in the corresponding correctness game. This may be done by using the oracles of the PSDVRS game to produce necessary PSDVRS's for the signing queries, and passing (i^*, j^*, μ^*) from $(i^*, \mathcal{R}^*, j^*, \mu^*)$ provided by $\mathcal{A}$ on to the PSDVRS challenger. Verification queries may be performed by passing PSDVRS signatures to the PSDVRS verification oracle, along with the proofs of provenance extracted from π. This generates an invalid PSDVRS with the distinguishing advantage of $\mathcal{A}$.

$$\mathsf{Adv}^{\text{COR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u) \leq \epsilon^{\mathsf{NIZK}}_{\text{ZK}} + Q_{\mathsf{Verify}} \cdot \epsilon^{\mathsf{NIZK}}_{\text{SIM-EXT}} + \epsilon^{\mathsf{PSDVRS}}_{\text{COR}}$$

Consistency. We prove the MDVRS scheme is consistent following Definition 3. We again proceed by defining a sequence of indistinguishable hybrids:

$\mathcal{H}_0$: Run $\mathsf{Experiment}^{\text{CON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$ as described in Figure 4. Observe that a winning adversary must provide a proof π^* as a part of σ^* which verifies.

$\mathcal{H}_1$: Modify $\mathcal{H}_0$ to simulate all proofs contained in signatures produced by the oracles.

$\mathcal{H}_2$: Same as $\mathcal{H}_1$, except use the knowledge extractor for the adversary to extract a valid witness for π^*, aborting the experiment if this fails. Furthmore, for each verification of the at most Q_{Verify} query extract the witness from the NIZK π if the signature was not produced by the signing oracle, aborting if extraction fails. The extracted witness $w = \{\pi_j\}_{j \in \mathcal{D}}$ must either contain proofs such that the RealSigVal is satisfied for all $j \in \mathcal{D}$ or each π_j either satisfies PubSigVal or VerSigVal.

$\mathcal{H}_3$: Same as $\mathcal{H}_2$, except abort if the extracted proofs satisfy RealSigVal. Let π_j be a proof which does not satisfy RealSigVal.

$\mathcal{H}_4$: Same as $\mathcal{H}_3$, except abort if π_j satisfies PubSigVal.

$\mathcal{H}_5$: Same as $\mathcal{H}_4$, except abort π_j satisfies VerSigVal.

In $\mathcal{H}_5$ the adversary can now no longer win. The extracted witness w for a wining adversary must be valid, otherwise the changes for $\mathcal{H}_2$ would cause an abort. This leads to a contradiction as such a witness must either contain proofs which all satisfy RealSigVal, excluded by $\mathcal{H}_3$, or have proofs which all satisfy either PubSigVal or VerSigVal. Hybrids $\mathcal{H}_4$ and $\mathcal{H}_5$ exclude the latter for one of the proofs.

We prove the sequence of hybrids above are in fact indistinguishable:

$\mathcal{H}_0 \approx \mathcal{H}_1$**:** These hybrids are indistinguishable by the zero knowledge property of the NIZK. An adversary distinguishing these hybrids may be reduced to an adversary with the same advantage in the zero knowledge game of the NIZK.

$\mathcal{H}_1 \approx \mathcal{H}_2$**:** Simulation extractability ensures that extraction for any single proof cannot fail with non-negligible probability. $\mathcal{H}_2$ aborts if the adversary produces a proof and statement which would win the simulation extractability game for one of the $(Q_{\mathsf{Verify}} + 1)$ proofs.

$\mathcal{H}_2 \approx \mathcal{H}_3$**:** An adversary producing proofs which all satisfy RealSigVal may be used to win the provable signing soundness game (Figure 9). As verification fails for one of the designated verifiers the PSDVRS directed toward them must not verify and may therefore be output with the corresponding proof to the challenger.

$\mathcal{H}_3 \approx \mathcal{H}_4$**:** The proof satisfying PubSigVal may be used along with the verifying PSDVRS to win the PubSigSim soundness game (Figure 12).

$\mathcal{H}_4 \approx \mathcal{H}_5$**:** The proof satisfying $\mathsf{VerSigVal}$ may be used along with the verifying PSDVRS to win the VerSigSim soundness game (Figure 15).

$$\begin{aligned}\mathsf{Adv}^{\text{CON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa,u) \leq \epsilon^{\mathsf{NIZK}}_{\text{ZK}} + (1+Q_{\mathsf{Verify}})\epsilon^{\mathsf{NIZK}}_{\text{SIM-EXT}} + \epsilon^{\mathsf{PSDVRS}}_{\text{ProvSig-sound}} \\ + \epsilon^{\mathsf{PSDVRS}}_{\text{PubSigSim-sound}} + \epsilon^{\mathsf{PSDVRS}}_{\text{VerSigSim-sound}}\end{aligned}$$

Unforgeability. We prove the MDVRS scheme described in Figure 16 is unforgeable (Definition 4). Consider the following hybrids:

$\mathcal{H}_0$: Run unforgeability game $\mathsf{Experiment}^{\text{EUF}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa,u)$ described in Figure 5.

$\mathcal{H}_1$: Same as $\mathcal{H}_0$, but replace all proofs returned in signing queries by simulated proofs. This is indistinguishable from the previous hybrid by the zero knowledge property of the NIZK.

$\mathcal{H}_2$: Same as $\mathcal{H}_1$, but for every verification query, extract the witness from the proof contained in the signature. Abort if extraction fails. Simulation extractability ensures that extraction for any single proof cannot fail with non-negligible probability.

An adversary winning the game in the hybrid $\mathcal{H}_1$ may be reduced to and adversary winning $\mathsf{Experiment}^{\text{EUF}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u)$ with the same advantage. To produce response to a $O\mathsf{Sign}$ query the required PSDVRS signatures from the oracle, and simulate the required proof. For a $O\mathsf{Verify}$ query the proof may be verified directly, then using the verification query in the PSDVRS game to verify the appropriate designated signature, using the extracted proof of provenance. All other oracle queries may be mapped directly to the corresponding queries of the PSDVRS game.

$$\mathsf{Adv}^{\text{EUF}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa,u) \leq \epsilon_{\text{ZK-NIZK}} + Q_{\mathsf{Verify}} \cdot \epsilon^{\mathsf{NIZK}}_{\text{SIM-EXT}} + \epsilon_{\text{EUF-PSDVRS}}$$

Off-The-Record. We prove the MDVRS scheme described in Figure 16 is off-the-record following Definition 5.

$\mathcal{H}_0$: Run $\mathsf{Experiment}^{\text{OTR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa,u)$ as described in Figure 6.

$\mathcal{H}_1$: Same as $\mathcal{H}_0$, except replace the honestly produced CRS with a simulated one, and simulate all proofs produced by the oracles.

$\mathcal{H}_2$: Same as $\mathcal{H}_1$, but for every verification query, extract the witness from the proof contained in the signature. Abort if extraction fails.

$\mathcal{H}_3$: In a series of hybrids replace verifier simulations with real signatures one-by-one. The number of hybrids in this sequence is bounded by the maximal number designated verifiers.

$\mathcal{H}_4$: In a series of hybrids replace public simulations with real signatures one-by-one. The length of this sequence of hybrids is bounded by the number of designated verifiers above. In final hybrid of this sequence σ_0 and σ_1 are identically distributed.

$\mathcal{H}_0 \approx \mathcal{H}_1$**:** These hybrids are indistinguishable by the zero knowledge property of the NIZK.

$\mathcal{H}_1 \approx \mathcal{H}_2$**:** Simulation extractability ensures that extraction for any single proof fails with probability less than $\epsilon^{\mathsf{NIZK}}_{\text{SIM-EXT}}$.

$\mathcal{H}_2 \approx \mathcal{H}_3$**:** Each pair of hybrids in this sequence are indistinguishable by verifier signature simulation indistinguishability. An adversary successfully distinguishing two hybrids may be used win the experiment $\mathsf{Experiment}^{\text{VerSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ with the same advantage using the challenge from the game in place of the PSDVRS which is being replaced. Note, a verifier simulation is indistinguishable even given access to the verifiers keys.

$\mathcal{H}_3 \approx \mathcal{H}_4$**:** Each pair in this sequence is indistinguishable by public signature simulation indistinguishability. In this case a winning adversary may be reduced to one winning $\mathsf{Experiment}^{\text{PubSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$. The off-the-record game ensures that this verifier is not corrupted, as required by the PubSigSim-Ind game.

$$\mathsf{Adv}^{\text{OTR}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u) \leq \epsilon^{\mathsf{NIZK}}_{\text{ZK}} + Q_{\mathsf{Verify}} \cdot \epsilon^{\mathsf{NIZK}}_{\text{SIM-EXT}} + u \cdot (\epsilon^{\mathsf{PSDVRS}}_{\text{VerSigSim-Ind}} + \epsilon^{\mathsf{PSDVRS}}_{\text{PubSigSimInd-Ind}})$$

Anonymity. We prove the MDVRS scheme described in Figure 16 is anonymous (Definition 6).

$\mathcal{H}_0$: Run $\mathsf{Experiment}^{\text{ANON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u)$ as described in Figure 7.

$\mathcal{H}_1$: Same as $\mathcal{H}_0$, except simulate all proofs produced by the oracles.

$\mathcal{H}_2$: Same as $\mathcal{H}_1$, but for every verification query, extract the witness from the proof contained in the signature. Abort if extraction fails.

$\mathcal{H}_3$: In a series of hybrids PSDVRS's from signer i^*_0 are replaced one-by-one by signatures from i^*_1.

$\mathcal{H}_0 \approx \mathcal{H}_1$**:** These hybrids are indistinguishable by the zero knowledge property of NIZK.

$\mathcal{H}_1 \approx \mathcal{H}_2$**:** Simulation extractability ensures that extraction for any single proof fails with probability less than $\epsilon^{\mathsf{NIZK}}_{\text{SIM-EXT}}$.

$\mathcal{H}_2 \approx \mathcal{H}_3$ Each pair of hybrids in this sequence are indistinguishable by the anonymity of the PSDVRS scheme. An adversary distinguishing these hybrids may be reduced to an adversary with the same advantage in $\mathsf{Experiment}^{\text{ANON}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$.

$$\mathsf{Adv}^{\text{ANON}}_{\mathsf{MDVRS},\mathcal{A}}(\kappa, u) \leq \epsilon^{\mathsf{NIZK}}_{\text{ZK}} + Q_{\mathsf{Verify}} \cdot \epsilon^{\mathsf{NIZK}}_{\text{SIM-EXT}} + u \cdot \epsilon^{\mathsf{PSDVRS}}_{\text{ANON}}$$

□

6 PSDVRS From the Discrete Logarithm Problem

We design a PSDVRS scheme based on the discrete logarithm problem, taking inspiration from the Designated Verifier Linkable Ring Signature scheme of Balla *et al.* [BBG+22]. Our approach follows the established paradigm of first devising an interactive identification scheme, and then transforming it to a signature scheme using the Fiat-Shamir transform [FS87]. We present a simple linear size proof, and later discuss how this may be improved to achieve $O(\log n)$ complexity.

6.1 Construction

Where relevant, we state randomness used by an algorithm explicitly as an extra parameter r, e.g. for a randomized algorithm $\mathsf{Setup}(1^\kappa)$, we may instead refer to $\mathsf{Setup}(1^\kappa; r)$ where Setup is now a deterministic function. For procedures and games with significant similarities, we will save space by presenting both variants in one figure, highlighting details which only appear in one of the games with *light* and *dark* gray.

A Publicly Verifiable OR-proof. Consider the following sigma protocol proving knowledge of the exponent for an element in the set $\{X_i\}_{i\in\mathcal{R}}$, which follows the blueprint of Cramer, Damgård, and Schoenmakers [CDS94].

This proof will be publicly verifiable once the Fiat-Shamir transform is applied. The prover $\mathcal{P}$ knows x such that $X_k = g^x$ and interacts with a public-coin verifier $\mathcal{V}$.

1. $\mathcal{P}$: Sample r and $\{c_i\}_{i\in\mathcal{R}\setminus\{k\}}$ uniformly in $\mathbb{Z}_q$. Compute the first message of the Sigma protocol as $A \leftarrow g^r \left(\prod_{i\in\mathcal{R}\setminus\{k\}} X_i^{c_i}\right)$. Send A to $\mathcal{V}$.
2. $\mathcal{V}$: Sample a uniformly random challenge c in $\mathbb{Z}_q$ and send it to $\mathcal{P}$.
3. $\mathcal{P}$: Let $c_k = c - \sum_{i\in\mathcal{R}\setminus\{k\}} c_i$. Compute, $z = r - c_k \cdot x$, and then send z along with $\{c_i\}_{i\in\mathcal{R}}$ to $\mathcal{V}$.
4. $\mathcal{V}$: The verifier checks, $A = g^z \left(\prod_{i\in\mathcal{R}} X_i^{c_i}\right)$ and $c = \sum_{i\in\mathcal{R}} c_i$ accepting if and only if both relations hold.

We will rely on the special-soundness and honest-verifier zero knowledge of this protocol. To allow the verifier to simulate signatures we simply add the verifier to $\mathcal{R}$.

Getting Public Simulation with a Designated Verifier. The previous protocol is unsuitable for our purposes, as it is publicly verifiable. To circumvent this we extend the public key of the verifier to include $V = g^v$, and replace the first message of the Sigma protocol with an El-Gamal style encryption of the same message. The prover's first message will now be $A_1 = g^{r_1}$ and $A_2 = \left(g^{r_2} \prod_{i\in\mathcal{R}\setminus\{k\}} X_i^{c_i}\right) V^{r_1}$. Challenge and response (with $r = r_2$) are computed as before. The verification, however, must be extended to include decryption using the secret exponent v, $A_1^{-v} A_2 = g^z \prod_{i\in\mathcal{R}} X_i^{c_i}$. Public simulation now becomes possible if DDH is hard in the group. We present the resulting designated verifier NIZKAoK in Figure 17.

Provable Signing. To enable proof of provenance for non publicly simulated signatures we add a commitment to the author's secret key. A signer (or verifier) may then prove in zero knowledge that the committed value matches their public key. We ensure it is hard to produce public simulations that are also valid signatures, by requiring they be produced using a different generator h for the same group. Given a public simulation which verifies, along with its proof of provenance, the discrete log of h with respect to g may be found.

We formalise the resulting signature scheme in Figure 18 and 19.

Zero Knowledge Friendly Provable Signing and Simulation. We are designing our PSDVRS scheme with the goal that it may be used to efficiently instantiate a MDVRS scheme. To this end, the predicates $\mathsf{RealSigVal}, \mathsf{VerSigVal}$ and $\mathsf{PubSigVal}$ be satisfied need to be efficiently provable in zero knowledge. With this in mind, it is desirable to avoid the need for generic proofs of general circuits, such as proving correct evaluation of cryptographic hash functions. Both Prove and PubSim require calls to H_q. However, the evaluated inputs (ϕ, A_1, A_2) are publicly known, allowing the hash outputs to be publicly computed, and then be treated as fixed constants in $\mathbb{Z}_q$ for the purposes of the proof. With this observation, proving $\mathsf{RealSigVal}, \mathsf{VerSigVal}$ and $\mathsf{PubSigVal}$ in zero knowledge only requires a combination of proofs of knowledge of exponents and arithmetic over $\mathbb{G}$ and $\mathbb{Z}_q$. Showing that proofs verify cannot be done directly as the verification algorithm requires access to the verifier's secret key. Instead, we include the randomness used to produce the proof in the proof of provenance, allowing the validity of the proof to follow from the perfect completeness of the proof system.

$\mathsf{Prove}(\phi, w)$

1 : **parse** $\phi = (\mathtt{vpk} = (X, V), \{X_i\}_{i\in\mathcal{R}}, \mathtt{aux})$
2 : **parse** $w = (k, x_k)$
3 : $r_1, r_2, \{c_i\}_{i\in\mathcal{R}\setminus\{k\}} \leftarrow^{\$} \mathbb{Z}_q$
4 : $A_1 \leftarrow g^{r_1}$ // First half of ElGamal ciphertext
5 : $A_2 \leftarrow V^{r_1} g^{r_2} \left(\prod_{i\in\mathcal{R}\setminus\{k\}} X_i^{c_i}\right)$ // Second half of ElGamal ciphertext
6 : $c_k = \mathsf{H}_q(\mathtt{pp}, \phi, A_1, A_2) - \left(\sum_{i\in\mathcal{R}\setminus\{k\}} c_i\right) \mod q$
7 : $z \leftarrow (r_2 - c_k \cdot x_k)$
8 : **return** $(A_1, A_2, z, \{c_i\}_{i\in\mathcal{R}})$

$\mathsf{Verify}(\phi, \pi, v)$

1 : **parse** $\phi = (\mathtt{vpk} = (X, V), \{X_i\}_{i\in\mathcal{R}}, \mathtt{aux})$
2 : **parse** $\pi = (A_1, A_2, z, \{c_i\}_{i\in\mathcal{R}})$
3 : **if** $A_1^{-v} A_2 \neq g^z \left(\prod_{i\in\mathcal{R}} X_i^{c_i}\right)$ **then return** 0
4 : **if** $\mathsf{H}_q(\mathtt{pp}, \phi, A_1, A_2) \neq \left(\sum_{i\in\mathcal{R}} c_i\right) \mod q$ **then return** 0
5 : **return** 1

$\mathsf{HVZKSim}(\phi, c, V)$

1 : **parse** $\phi = (\mathtt{vpk} = (X, V), \{X_i\}_{i\in\mathcal{R}}, \mathtt{aux})$
2 : $\{c_i\}_{i\in[\mathcal{R}]} \leftarrow^{\$} \mathbb{Z}_q$ subject to $\sum_{i\in\mathcal{R}} c_i = c \mod q$
3 : $z, r \leftarrow^{\$} \mathbb{Z}_q$
4 : $A_1 \leftarrow g^r$
5 : $A_2 \leftarrow V^r \cdot g^z \left(\prod_{i\in\mathcal{R}} X_i^{c_i}\right)$
6 : **return** $(A_1, A_2, z, \{c_i\}_{i\in\mathcal{R}})$

$\mathsf{PubSim}(\phi)$

1 : **parse** $\phi = (\mathtt{vpk} = (X, V), \{X_i\}_{i\in\mathcal{R}}, \mathtt{aux})$
2 : $r_1 \leftarrow^{\$} \mathbb{Z}_q;\ r_2 \leftarrow^{\$} \mathbb{Z}_q \setminus \{0\}$
3 : $A_1 \leftarrow g^{r_1};\ A_2 \leftarrow h^{r_2}$
4 : $c = \mathsf{H}_q(\mathtt{pp}, \phi, A_1, A_2)$
5 : $(A_1', A_2', z, \{c_i\}_{i\in\mathcal{R}}) \leftarrow \mathsf{HVZKSim}(\phi, c, V)$ // First prover message is ignored
6 : **return** $(A_1, A_2, z, \{c_i\}_{i\in\mathcal{R}})$

Figure 17: Our publicly simulatable designated verifier NIZKAoK construction. We assume all algorithms have access to a public string of the form $(\mathbb{G}, q, g, h) \leftarrow \mathit{GroupGen}(1^\kappa)$, where g, h are two independent generators of $\mathbb{G}$.

6.2 Security Statement

Before we prove Theorem 2 we will prove security of the Sigma protocol from Section 6.1.

Lemma 1. *The Σ-protocol in Section 6.1 is special-sound for the relation,*

$$\mathscr{R}^{\Sigma}_{\mathtt{pp}} = \left\{ \begin{array}{l} \phi = (\mathcal{R}, \{X_i\}_{i\in\mathcal{R}}, \mathtt{aux}) \\ w = (\rho, \mathcal{S}, \{\rho_i\}_{i\in\mathcal{S}}) \end{array} \middle| \mathcal{S} \subset \mathcal{R},\ g^\rho = \prod_{i\in\mathcal{S}} X_i^{\rho_i},\ \forall i \in \mathcal{S}\ \rho_i \neq 0 \right\},$$

and has special honest verifier zero-knowledge.

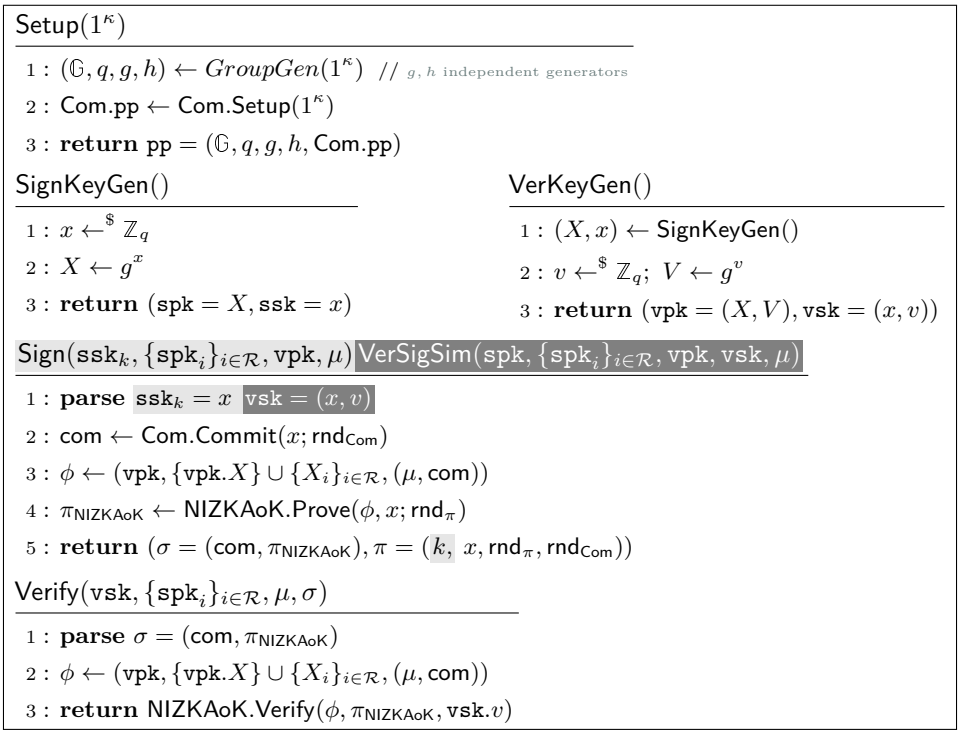

Figure 18: Our concrete construction of PSDVRS that relies on NIZKAoK described in Figure 17. For the remaining algorithms see Figure 19.

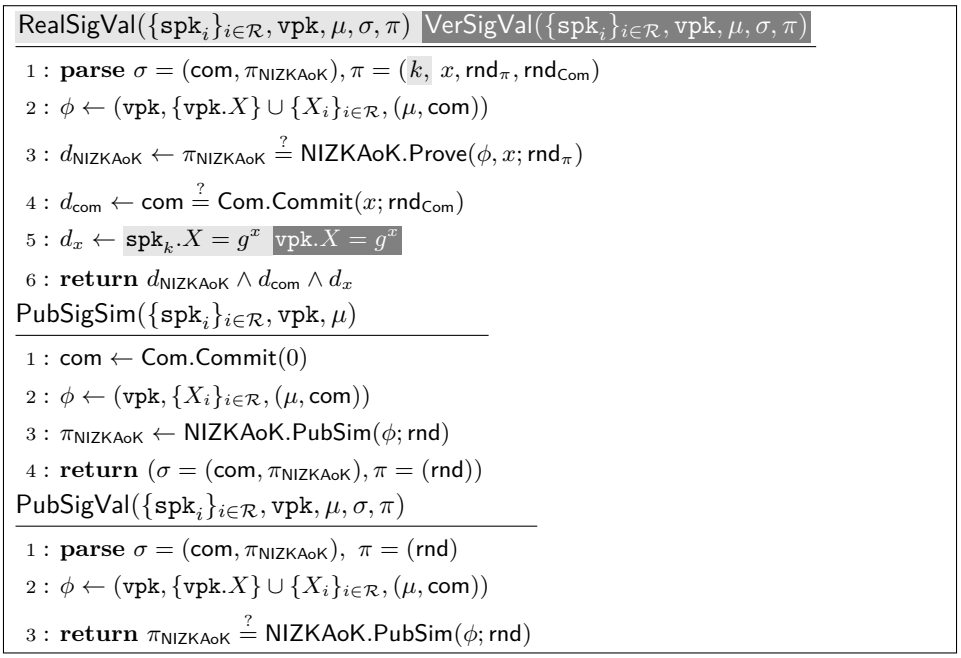

Figure 19: Our concrete PSDVRS construction (remaining algorithms).

Proof. The relation $\mathscr{R}_{\mathsf{pp}}^{\Sigma}$ is relaxed compared to the requirement we make of the honest prover but is sufficient for our needs.

Special soundness. Assume we have two accepting transcripts, with the same first message and distinct challenges $(A, c, (z, \{c_i\}_{i\in\mathcal{R}})), (A, c', (z', \{c'_i\}_{i\in\mathcal{R}}))$. By assumption,

$$g^z A = \prod_{i\in\mathcal{R}} X_i^{c_i}, \quad g^{z'} A = \prod_{i\in\mathcal{R}} X_i^{c'_i}$$

Thus,

$$g^{z-z'} = \prod_{i\in\mathcal{R}} X_i^{c_i - c'_i}.$$

As $c \neq c'$, we know $\{c_i\}_{i\in\mathcal{R}} \neq \{c'_i\}_{i\in\mathcal{R}}$. Let $\mathcal{S}$ be the non-empty set of indices i where $c_i \neq c'_i$, then

$$g^{z-z'} = \prod_{i\in\mathcal{S}} X_i^{c_i - c'_i}.$$

We have now obtained a witness for $\mathscr{R}^{\Sigma}_{\mathsf{pp}}$, which would allow finding the discrete log of a group element X_i with $i \in \mathcal{S}$, if the remaining X_j for $j \in \mathcal{S} \setminus \{i\}$ are known powers of g.

(Special) HVZK. To simulate signatures we will require HVZK for our protocol. Consider the simulator, which given a challenge c does the following:

- Sample c_i uniformly in $\mathbb{Z}_q$ for $i \in \mathcal{R}$ subject to $\sum_{i\in\mathcal{R}'} c_i = c$.
- Sample z uniformly in $\mathbb{Z}_q$.
- Let $A = g^{-z} \prod_{i\in\mathcal{R}} X_i^{c_i}$.

The distribution of the challenges $\{c_i\}_{i\in\mathcal{R}}$ is clearly identical between the real and simulated transcripts. In the real transcripts $z = r + c_i x_i$ is also uniform in $\mathbb{Z}_q$ as r is uniform in $\mathbb{Z}_q$. For fixed z and $\{c_i\}_{i\in\mathcal{R}}$ the first message A is uniquely determined. Note, this simulation is perfect. □

Special-soundness implies knowledge-soundness for the Fiat-Shamir transform of our proof [AFK23]. It is clear that the protocol remains special-sound if the statement is expanded to include the verifier decryption key after the transformation to Section 6.1. To achieve adaptive knowledge soundness, the prover must be bound to a choice of statement while producing a proof. This is done by including the statement as an input to the random oracle. The prover does not have access to the verifier decryption key, preventing it from inputting this to the random oracle. However, the verifier public key serves as a perfectly binding commitment to the verifier key; it is therefore sufficient to include the verifier public key in place of the secret key.

The HVZK simulator may easily be extended to allow the modifications in Section 6.1, encrypting the first message as in the real protocol. For the resulting simulator see HVZKSim in Figure 17.

We now proceed to prove the security of our PSDVRS construction in Section 6.

Theorem 2. *The PSDVRS scheme described in Section 6 is secure in the programmable random oracle model (pROM) according to Definition 19, assuming the security of the commitment scheme and the hardness of the decisional Diffie-Hellman (DDH) and discrete logarithm problem in the cyclic group $\mathbb{G}$ of order q with generators g, h. Let ϵ_{DDH} and ϵ_{DLOG} bound the advantage in solving the aforementioned problems. Consider the upper bounds Q_{Sign} and Q_{H} on the number of queries made by the adversary to the signing and random oracle respectively. Fix $\epsilon^{\mathsf{Com}}_{\text{HIDING}}$ as a bound on the advantage of an adversary breaking hiding*

for the commitment scheme. Further, let $\epsilon_{\text{SIM}} = Q_{\mathsf{H}}/q$ *and* $\epsilon_{\text{KE}} \leq (Q_{\mathsf{H}}+1)/q$*. Then the scheme is then secure with the following advantages. Correctness:*

$$\mathsf{Adv}^{\text{COR}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u) = \mathsf{Adv}^{\text{ProvSig-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u) = 0,$$

$$\mathsf{Adv}^{\text{PubSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u) = \mathsf{Adv}^{\text{VerSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u) = 0;$$

soundness:

$$\mathsf{Adv}^{\text{ProvSig-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u) = 0,\ \mathsf{Adv}^{\text{VerSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u) \leq u \cdot \epsilon_{\text{DLog}},$$

$$\mathsf{Adv}^{\text{PubSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u) \leq \epsilon_{\text{DLog}};$$

signature simulation indistinguishability:

$$\mathsf{Adv}^{\text{PubSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u) \leq \epsilon^{\mathsf{Com}}_{\text{HIDING}} + \epsilon_{\text{DDH}} + 1/q + \mathsf{Adv}^{\text{PubSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u),$$

$$\mathsf{Adv}^{\text{VerSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u) \leq \epsilon^{\mathsf{Com}}_{\text{HIDING}} + 2\epsilon_{\text{SIM}};$$

and unforgeability, signer anonymity

$$\mathsf{Adv}^{\text{EUF}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u) \leq Q_{\mathsf{Sign}}\epsilon_{\text{SIM}} + \epsilon_{\text{KE}} + u \cdot \epsilon_{\text{DLog}},$$

$$\mathsf{Adv}^{\text{ANON}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa,u) \leq \epsilon^{\mathsf{Com}}_{\text{HIDING}} + 2\epsilon_{\text{SIM}}.$$

Secure erasure is assumed for the case of adaptive corruptions.

Proof. We prove security of our PSDVRS construction described in Figures 17, 18 and 19. Recall u is an upper bound on the total number of registered signers and verifiers.

Oracles. When we will need to simulate the signing oracle we will do the following. For $O\mathsf{Sign}(k,\mathcal{R},j,\mu)$

- Compute $\mathsf{com} \leftarrow \mathsf{Com.Commit}(0)$
- Let $\phi = (\mathtt{vpk}_j, \{\mathtt{vpk}_j.Z\} \cup \{X_i\}_{i\in\mathcal{R}}, (\mu, \mathsf{com}))$
- Sample $c \leftarrow^{\$} \mathbb{Z}_q$ and compute $\pi_{\mathsf{NIZK}} \leftarrow \mathsf{HVZKSim}(\phi, c, \mathtt{vpk}.V)$
- Reprogram the random oracle such that $\mathsf{H}(\phi, A_1, A_2) = c$, aborting if (ϕ, A_1, A_2) was queried previously.

As $\mathsf{HVZKSim}$ perfectly simulates transcripts an adversary will only be able to distinguish simulated signatures if a previously queried input to the oracle is reprogrammed. For a single signing query A_1 will be independently uniform in the group $\mathbb{G}$, if the adversary has previously made Q_{H} queries to H then the probability A_1 coincides with any previously queried transcript is at most $\epsilon_{\text{SIM}} = Q_{\mathsf{H}}/q$, where q is the order of $\mathbb{G}$.[2] By a union bound across the signing queries, the probability of aborting is at most $Q_{\mathsf{Sign}}\epsilon_{\text{SIM}}$. Note, the signature is also randomised by the commitment, which is included in the statement; if the commitments were instantiated with Pedersen commitments [Ped92] the probability of aborting would in fact be bounded by $\epsilon_{\text{SIM}} = Q_{\mathsf{H}}/q^2$. Randomising the random oracle input with an additional salt would allow for an even smaller ϵ_{SIM}.

Correctness. Perfect correctness follows by inspection of the scheme.

[2] For convenience, we will also count the indirect random oracle queries made when querying $O\mathsf{Sign}$ towards Q_{H}.

Existential Unforgeability. We prove unforgeability (Definition 9, Figure 5) of our PSDVRS scheme by a reduction to the discrete logarithm problem.

Consider an adversary $\mathcal{A}$ in the experiment $\mathsf{Experiment}^{\text{EUF}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ as described in Figure 5 with advantage ϵ. Our high level goal is to extract a witness for the NIZK contained in the signature, relying on the knowledge soundness of the proof system. Specifically, we wish to rely on the adaptive knowledge soundness [AFK23, Definition 10] of special sound protocols under the Fiat-Shamir transform. However, our current adversary has access to a number of oracles, making it incompatible with the standard extraction techniques. To address this we first construct an intermediate adversary which runs $\mathcal{A}$ internally and simulates access to the oracles.

More formally, consider $\mathcal{B}$ that plays the role as the challenger in the $\mathsf{Experiment}^{\text{EUF}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$, running $\mathcal{A}$ internally and outputting π_{NIZK} contained in the forgery σ^*. Queries to the random oracle are simply forwarded by $\mathcal{B}$ to the real oracle. Note, $\mathcal{B}$ outputs a valid proof with probability ϵ (the advantage of $\mathcal{A}$).

We wish to reduce to discrete log, to this end we may replace $\mathcal{B}$ with $\mathcal{B}'$ that receives a discrete logarithm challenge X. The new adversary $\mathcal{B}'$ follows the instructions of the challenger in $\mathsf{Experiment}^{\text{EUF}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$, except for a random user index $i \in [u]$, where it sets the signer public key to X, and replaces responses to $O\mathsf{Sign}(i, \cdot, \cdot, \cdot)$ with simulated signatures. If $\mathcal{A}$ corrupts i then $\mathcal{B}'$ must abort. $\mathcal{B}'$ outputs π_{NIZK} as $\mathcal{B}$. Note, the random oracle is only reprogrammed in positions which may not be used by a valid forgery. Let c be the number of users $\mathcal{A}$ has corrupted, then $\mathcal{B}'$ will output a valid proof (and not abort) with probability at least $(\frac{u-c}{u})\epsilon - Q_{\mathsf{Sign}}\epsilon_{\text{SIM}}$.

By the knowledge soundness of $\mathsf{NIZKAoK}$ there must be an efficient extractor $\mathcal{E}^{\mathcal{B}'}$ which when given black-box rewind access to $\mathcal{B}'$ extracts a valid witness, except with knowledge error ϵ_{KE}. If the extractor is successful $(\phi = (\mathtt{vpk}_j, \{\mathtt{vpk}_j.Z\} \cup \{X_i\}_{i\in\mathcal{R}}, (\mu, \mathsf{com})), w = (\rho, \mathcal{S}, \{\rho_i\}_{i\in\mathcal{S}})) \in \mathscr{R}_\Sigma$ is obtained, note the distribution of ϕ produced by the extractor is the same as the distribution of statements produced by the prover. With probability at least $1/(u-c)$, the identity i will be in $\mathcal{S}$, allowing the discrete logarithm of X to be efficiently computed. The resulting extractor succeeds with probability,

$$\frac{1}{u-c}\left(\left(\frac{u-c}{u}\right)\epsilon - Q_{\mathsf{Sign}}\epsilon_{\text{SIM}} - \epsilon_{\text{KE}}\right) \leq \frac{\epsilon}{u} - Q_{\mathsf{Sign}}\epsilon_{\text{SIM}} - \epsilon_{\text{KE}}.$$

Thus, we may conclude

$$\mathsf{Adv}^{\text{EUF}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) \leq Q_{\mathsf{Sign}}\epsilon_{\text{SIM}} + \epsilon_{\text{KE}} + u \cdot \epsilon_{\text{DLOG}}.$$

By [AFK23], for a special-sound protocol $\epsilon_{\text{KE}} \leq (Q_{\mathsf{H}} + 1)/q$.

Signer Anonymity. We prove signer anonymity (Definition 10, Figure 7) by reducing to the hiding property of the commitment scheme and HVZK of our NIZK.

$\mathcal{H}_0$: Run $\mathsf{Experiment}^{\text{ANON}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ as described in Figure 7.

$\mathcal{H}_1$: The same as $\mathcal{H}_0$, but for $b = 0$ rather than committing to $\mathtt{ssk}_{i_0}$ commit to $\mathtt{ssk}_{i_1}$.

$\mathcal{H}_2$: The same as $\mathcal{H}_1$ but for $b = 0$ rather than signing honestly, use $\mathsf{HVZKSim}$ to simulate the signature, reprogramming the random oracle as described previously.

$\mathcal{H}_3$: The same as $\mathcal{H}_2$ but for $b = 0$ rather than using $\mathsf{HVZKSim}$ to simulate the signature when $b = 0$, now sign using $\mathtt{ssk}_{i_1}$. For $b \in \{0, 1\}$ the view of the adversary is now identical.

We now prove indistinguishability of the hybrids described above:

$\mathcal{H}_0 \approx \mathcal{H}_1$**:** An adversary distinguishing these hybrids may be reduced to an adversary winning the hiding game of the commitment scheme with the same advantage.

$\mathcal{H}_1 \approx \mathcal{H}_2 \approx \mathcal{H}_3$**:** By the perfect simulation of $\mathsf{HVZKSim}$ these cases may only be distinguished when the oracle cannot be reprogrammed, giving advantage ϵ_{SIM}.

$$\mathsf{Adv}^{\text{ANON}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) \leq \epsilon^{\mathsf{Com}}_{\text{HIDING}} + 2\epsilon_{\text{SIM}}$$

Public Signature Simulation Indistinguishability. We prove public signature simulation indistinguishability (Definition 13, Figure 10) by reduction to the hiding property of the commitment scheme and the decisional Diffie-Hellman (DDH) problem.

$\mathcal{H}_0$: Run $\mathsf{Experiment}^{\text{PubSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u)$ as described in Figure 10.

$\mathcal{H}_1$: The same as $\mathcal{H}_0$, but in Sign replace the commitment by a commitment to 0.

$\mathcal{H}_2$: The same as $\mathcal{H}_1$, but during Sign replace π_{NIZK} with a simulated proof produced by PubSim. The distributions for $b \in \{0, 1\}$ are now identical.

We now prove indistinguishability of the hybrids described above:

$\mathcal{H}_0 \approx \mathcal{H}_1$**:** An adversary distinguishing these hybrids may be reduced to an adversary breaking hiding for the commitment scheme with the same advantage.

$\mathcal{H}_1 \approx \mathcal{H}_2$**:** Consider a DDH challenger giving $(A = g^a, B = g^b, C = g^c)$ where c is either uniformly independent in $\mathbb{Z}_q$ or $c = ab$. An adversary distinguishing $\mathcal{H}_1$ and $\mathcal{H}_2$ may be reduced to an adversary for the DDH. The reduction chooses a random index $j \in [u]$ and sets $\mathsf{vpk}.V = A$. If $j^* \neq j$ the reduction aborts. As the reduction no longer has access to the verifier key, it can no longer simulate the verification oracle directly. However, the verification oracle requires the adversary to provide a proof satisfying one of $\mathsf{RealSigVal}, \mathsf{VerSigVal}$ or $\mathsf{PubSigVal}$. When either $\mathsf{RealSigVal}$ or $\mathsf{VerSigVal}$ are statisfied, $O\mathsf{Verify}$ may output 1, as the proof is guaranteed to verify by completeness of the NIZK. If $\mathsf{PubSigVal}$ is satisfied, $O\mathsf{Verify}$ may output 0; this response will only be wrong when the adversary provides a query which would win public signature simulation soundness, allowing us to bound this event by the corresponding advantage.

To produce the challenge signature, the reduction sets $A_1 = B$ and $A_2 = C \cdot g^{r_2} \prod_{i \in \mathcal{R} \setminus \{k\}} X_i^{c_i}$. In the case $c = ab$ this clearly gives the same distribution as regular signing. Alternatively, if c is uniform in $\mathbb{Z}_q$ the distribution is indistinguishable from that produced by PubSim if the check for $r_2 = 0$ is not made, this follows as A_2 is uniform in $\mathbb{G}$ independent of A_1. The statistical distance to the distribution where $r_2 \neq 0$ is ensured is $1/q$.

$$\mathsf{Adv}^{\text{PubSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) \leq \epsilon^{\mathsf{Com}}_{\text{HIDING}} + \epsilon_{\text{DDH}} + 1/q + \mathsf{Adv}^{\text{PubSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u).$$

Public Signature Simulation Correctness. (Definition 14, Figure 11) Follows by inspection, $\mathsf{Adv}^{\text{PubSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = 0$.

Public Signature Simulation Soundness. We prove public signature simulation soundness (Definition 15, Figure 12) by reduction to the discrete logarithm problem. As $\mathsf{PubSigVal}$ outputs 1, the adversary outputs randomness giving r where $A_1 = g^r$ and $A_2 = h^r$. As Verify also outputs 1, we also know

$$A_1^{-v} A_2 = g^z \prod_{i \in \mathcal{R}} Z_i^{c_i}.$$

The reduction knows the randomness used to generate all keys Z_i, allowing σ to be computed where

$$h^r = g^\sigma = g^{rv} g^z \prod_{i \in \mathcal{R}} Z_i^{c_i}.$$

By construction r cannot be 0, and therefore $r^{-1}\sigma \in \mathbb{Z}_q$ is a discrete logarithm solution,

$$\mathsf{Adv}^{\textsc{PubSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) \leq \epsilon_{\textsc{DLog}}.p$$

Verifier Signature Simulation Indistinguishability. (Definition 16, Figure 13) This follows by the same argumentation as for signer anonymity.

$$\mathsf{Adv}^{\textsc{VerSigSim-Ind}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) \leq \epsilon^{\mathsf{Com}}_{\textsc{HIDING}} + 2\epsilon_{\textsc{SIM}}.$$

Verifier Signature Simulation Correctness. (Definition 17, Figure 14) Follows by inspection, $\mathsf{Adv}^{\textsc{VerSigSim-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = 0$.

Verifier Signature Simulation Soundness. (Definition 18, Figure 15) An adversary winning this game may be reduced to an adversary solving the discrete logarithm problem. The reduction proceeds as follows. First it selects a random index $j \in [u]$, for this index it sets X to be the discrete log challenge. Note, the challenger can still provide a verification oracle for verifier j, as it may still know v where $V = g^v$. If the adversary chooses $j^* = j$, and wins the game it must have provided a discrete logarithm solution x such that $X = g^x$.

$$\mathsf{Adv}^{\textsc{VerSigSim-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) \leq u \cdot \epsilon_{\textsc{DLog}}.$$

Provable Signing Correctness. (Definition 11, Figure 8) Follows by inspection, $\mathsf{Adv}^{\textsc{ProvSig-Cor}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = 0$.

Provable Signing Soundness (Definition 12, Figure 9). For $\mathsf{RealSigVal}$ to output 1, d_{NIZK} must also be 1. By perfect completeness of NIZK it follows that the signature must verify, $\mathsf{Adv}^{\textsc{ProvSig-Sound}}_{\mathsf{PSDVRS},\mathcal{A}}(\kappa, u) = 0$. □

6.3 Further Optimisations

Attema *et al.* [ACF21] show how compressed sigma protocol theory [AC20] may be applied to achieve partial proofs of knowledge [CDS94] for discrete logarithms. We are interested in their 1-out-of-n proof, which has size and rounds logarithmic in n. In acordance with the compressed sigma protocol paradigm, their protocol is defined with respect to an initial sigma protocol Σ_0, known as the pivot, having a constant size commitment in the first round and a large (linear-size) opening in the final round. To reduce communication complexity, the prover proves knowledge of the opening rather than sending it directly. This proof of knowledge is instantiated by the use of a second sigma protocol Σ_1, with an opening that is smaller than the witness (the opening of Σ_0) by a constant fraction. In fact if this second protocol further allows proving knowledge of its own openings, the final message may be repeatedly replaced by an invocation Σ_1 until the opening reaches constant size. The resulting protocol satisfies a notion of special soundness generalised to multiple rounds. Note, it is sufficient for only Σ_0 to have HVZK for this composed protocol to be HVZK, as the opening of the pivot does not need to be hidden. In our signature scheme we consider the non interactive case; Attema *et al.* [AFK23] demonstrate that multi-round special-sound protocols are knowledge sound under the Fiat-Shamir transform [FS87].

We observe that this proof of [ACF21] can be transformed to fit the designated verifier setting by encrypting the first prover message as we did in Section 6.1. Public simulation still follows as the adversary cannot check if the challenge used in simulation corresponds to the first message. The recursive proofs of knowledge may be applied honestly to the third round message. In Figure 20 we show how our NIZKAoK (Figure 17) may be modified to use the proof of [ACF21] for the relation $\mathscr{R}_{\text{PARTIAL}}$. Substituting this for NIZKAoK in the PSDVRS construction (Figure 18, Figure 19) yeilds a scheme with signature sizes logarithmic in n. For the sake of self-containment recall the relation

$$\mathscr{R}_{\text{PARTIAL}} = \left\{ \begin{array}{l} \phi = (g, X_1, \dots X_n \in \mathbb{G}, k \in \{1, \dots, n\}, \mathtt{aux}), \\ w = (\mathcal{S} \subset [n], \mathbf{x} \in \mathbb{Z}_q^n) \end{array} \middle| \begin{array}{l} |\mathcal{S}| = k, \\ \forall i \in \mathcal{S}, X_i = g^{\mathbf{x}_i} \end{array} \right\},$$

from [ACF21], where $\mathbf{x}_i$ is the ith entry of the vector $\mathbf{x}$. For 1-of-n discrete logarithms in an order q cyclic group $\mathbb{G}$, the proof will contain $(4\lceil\log_2(2n)\rceil - 5)$ elements of $\mathbb{G}$ and 4 elements of $\mathbb{Z}_q$ [ACF21, Theorem 6] counting only prover messages due to Fiat-Shamir. Our final signature size will include the additional cost of encrypting the first message of Σ_0 and the commitment to the secret key from the sign procedure.

$\mathsf{Prove}(\phi, w)$

1 : **parse** $\phi = (\mathtt{vpk} = (X, \mathsf{PKE.vpk}), \{X_i\}_{i\in[n]}, \mathtt{aux})$
2 : **parse** $w = (k, x_k)$
3 : $\mathbf{x} \in \mathbb{Z}_q^n$ s.t. $\mathbf{x}_k = x_k$, and $\mathbf{x}_i = 0$ for $i \neq k$
  // Proving $((g, X_1, \dots X_n, 1), (\{k\}, \mathbf{x})) \in \mathscr{R}_{\text{PARTIAL}}$
4 : $(a_1, a_2 \dots a_\mu) \leftarrow \Pi_{\text{PARTIAL}}.\mathsf{Prove}((g, X_1, \dots X_n, 1, \mathtt{aux}), (\{k\}, \mathbf{x}))$
5 : $\sigma \leftarrow \mathsf{PKE.Enc}(\mathsf{PKE.vpk}, a_1)$
6 : **return** $(\sigma, a_2 \dots a_\mu)$

$\mathsf{Verify}(\phi, \pi, \mathsf{PKE.vsk})$

1 : **parse** $\phi = (\mathtt{vpk} = (X, \mathsf{PKE.vpk}), \{X_i\}_{i\in[n]}, \mathtt{aux})$
2 : **parse** $\pi = (\sigma, a_2 \dots a_\mu)$
3 : $a_1 \leftarrow \mathsf{PKE.Dec}(\mathsf{PKE.vsk}, \sigma)$
4 : **return** $\Pi_{\text{PARTIAL}}.\mathsf{Verify}((g, X_1, \dots X_n, 1, \mathtt{aux}), (a_1, \dots, a_\mu))$

$\mathsf{PubSim}(\phi)$

1 : **parse** $\phi = (\mathtt{vpk} = (X, \mathsf{PKE.vpk}), \{X_i\}_{i\in\mathcal{R}}, \mathtt{aux})$
  // Simulate as in [ACF21].
2 : $(a_1, a_2, \dots, a_\mu) \leftarrow \Pi_{\text{PARTIAL}}.\mathsf{HVZKSim}((g, X_1, \dots X_n, 1, \mathtt{aux}))$
3 : $\sigma \leftarrow \mathsf{PKE.Enc}(\mathsf{PKE.vpk}, a_1)$
4 : **return** $(\sigma, a_2 \dots a_\mu)$

Figure 20: Replacing our publicly simulatable designated verifier NIZKAoK with Π_{PARTIAL} from [ACF21]. We consider the non interactive variant of Π_{PARTIAL} after the application of the Fiat-Shamir transform, we omit all verifier messages as these may be recomputed using the random oracle. Note, the publicly simulated transcripts will not verify with respect to the random oracle, however, this is not a problem as a_1 would be needed to query the random oracle to observe this. More specifically due to the structure of Π_{PARTIAL} only first challenge would not be correctly computed, while all subsequent challenges are.

References

[AC20] Thomas Attema and Ronald Cramer. Compressed Σ-protocol theory and practical application to plug & play secure algorithmics. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part III*, volume 12172 of *LNCS*, pages 513–543. Springer, Cham, August 2020. doi:10.1007/978-3-030-56877-1_18.

[ACF21] Thomas Attema, Ronald Cramer, and Serge Fehr. Compressing proofs of k-out-of-n partial knowledge. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part IV*, volume 12828 of *LNCS*, pages 65–91, Virtual Event, August 2021. Springer, Cham. doi:10.1007/978-3-030-84259-8_3.

[AFK23] Thomas Attema, Serge Fehr, and Michael Klooß. Fiat-shamir transformation of multi-round interactive proofs (extended version). *Journal of Cryptology*, 36(4):36, October 2023. doi:10.1007/s00145-023-09478-y.

[AHAN+22] Diego F. Aranha, Mathias Hall-Andersen, Anca Nitulescu, Elena Pagnin, and Sophia Yakoubov. Count me in! Extendability for threshold ring signatures. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part II*, volume 13178 of *LNCS*, pages 379–406. Springer, Cham, March 2022. doi:10.1007/978-3-030-97131-1_13.

[BBG+22] Danai Balla, Pourandokht Behrouz, Panagiotis Grontas, Aris Pagourtzis, Marianna Spyrakou, and Giannis Vrettos. Designated-verifier linkable ring signatures with unconditional anonymity. In *Algebraic Informatics: 9th International Conference, CAI 2022, Virtual Event, October 27–29, 2022, Proceedings*, pages 55–68. Springer, 2022. doi:10.1007/978-3-031-19685-0_5.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *20th ACM STOC*, pages 103–112. ACM Press, May 1988. doi:10.1145/62212.62222.

[BGK+22] Pourandokht Behrouz, Panagiotis Grontas, Vangelis Konstantakatos, Aris Pagourtzis, and Marianna Spyrakou. Designated-verifier linkable ring signatures. In *Information Security and Cryptology–ICISC 2021: 24th International Conference, Seoul, South Korea, December 1–3, 2021, Revised Selected Papers*, pages 51–70. Springer, 2022. doi:10.1007/978-3-031-08896-4_3.

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 416–432. Springer, Berlin, Heidelberg, May 2003. doi:10.1007/3-540-39200-9_26.

[BKM06] Adam Bender, Jonathan Katz, and Ruggero Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. In Shai Halevi and Tal Rabin, editors, *TCC 2006*, volume 3876 of *LNCS*, pages 60–79. Springer, Berlin, Heidelberg, March 2006. doi:10.1007/11681878_4.

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In *ASIACRYPT 2001, volume 2248 of LNCS. Springer, Berlin, Heidelberg*, pages 514–532, 2001. doi:10.1007/3-540-45682-1_30.

[BSS02] Emmanuel Bresson, Jacques Stern, and Michael Szydlo. Threshold ring signatures and applications to ad-hoc groups. In Moti Yung, editor, *CRYPTO 2002*,

volume 2442 of *LNCS*, pages 465–480. Springer, Berlin, Heidelberg, August 2002. doi:10.1007/3-540-45708-9_30.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 174–187. Springer, Berlin, Heidelberg, August 1994. doi:10.1007/3-540-48658-5_19.

[Cho06] Sherman S. M. Chow. Identity-based strong multi-designated verifiers signatures. In *European Public Key Infrastructure Workshop*, 2006. doi:10.1007/11774716_23.

[Cho08] Sherman S. M. Chow. Multi-designated verifiers signatures revisited. *International Journal of Network Security*, 7:348–357, 2008. doi:10.6633/IJNS.200811.7(3).06.

[CK21] Matteo Campanelli and Hamidreza Khoshakhlagh. Succinct publicly-certifiable proofs - or, can a blockchain verify a designated-verifier proof? In Avishek Adhikari, Ralf Küsters, and Bart Preneel, editors, *INDOCRYPT 2021*, volume 13143 of *LNCS*, pages 607–631. Springer, Cham, December 2021. doi:10.1007/978-3-030-92518-5_27.

[CLHY05] Sherman SM Chow, Richard WC Lui, Lucas CK Hui, and Siu-Ming Yiu. Identity based ring signature: Why, how and what next. In *Public Key Infrastructure: Second European PKI Workshop: Research and Applications, EuroPKI 2005, Canterbury, UK, June 30-July 1, 2005, Revised Selected Papers 2*, pages 144–161. Springer, 2005. doi:10.1007/11533733_10.

[Cv91] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–265. Springer, Berlin, Heidelberg, April 1991. doi:10.1007/3-540-46416-6_22.

[DHM+20] Ivan Damgård, Helene Haagh, Rebekah Mercer, Anca Nitulescu, Claudio Orlandi, and Sophia Yakoubov. Stronger security and constructions of multi-designated verifier signatures. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 229–260. Springer, Cham, November 2020. doi:10.1007/978-3-030-64378-2_9.

[DKNS04] Yevgeniy Dodis, Aggelos Kiayias, Antonio Nicolosi, and Victor Shoup. Anonymous identification in ad hoc groups. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 609–626. Springer, Berlin, Heidelberg, May 2004. doi:10.1007/978-3-540-24676-3_36.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Berlin, Heidelberg, August 1987. doi:10.1007/3-540-47721-7_12.

[FS07] Eiichiro Fujisaki and Koutarou Suzuki. Traceable ring signature. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *PKC 2007*, volume 4450 of *LNCS*, pages 181–200. Springer, Berlin, Heidelberg, April 2007. doi:10.1007/978-3-540-71677-8_13.

[GKO+23] Chaya Ganesh, Yashvanth Kondi, Claudio Orlandi, Mahak Pancholi, Akira Takahashi, and Daniel Tschudi. Witness-succinct universally-composable SNARKs. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part II*, volume 14005 of *LNCS*, pages 315–346. Springer, Cham, April 2023. doi:10.1007/978-3-031-30617-4_11.

[GM17] Jens Groth and Mary Maller. Snarky signatures: Minimal signatures of knowledge from simulation-extractable SNARKs. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part II*, volume 10402 of *LNCS*, pages 581–612. Springer, Cham, August 2017. doi:10.1007/978-3-319-63715-0_20.

[Jou04] Antoine Joux. A one round protocol for tripartite Diffie–Hellman. *Journal of Cryptology*, 17(4):263–276, September 2004. doi:10.1007/s00145-004-0312-y.

[JSI96] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. Designated verifier proofs and their applications. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 143–154. Springer, Berlin, Heidelberg, May 1996. doi:10.1007/3-540-68339-9_13.

[LSMP07] Yong Li, Willy Susilo, Yi Mu, and Dingyi Pei. Designated verifier signature: definition, framework and new constructions. In *Ubiquitous Intelligence and Computing: 4th International Conference, UIC 2007, Hong Kong, China, July 11-13, 2007. Proceedings 4*, pages 1191–1200. Springer, 2007. doi:10.1007/978-3-540-73549-6_116.

[LV04] Fabien Laguillaumie and Damien Vergnaud. Multi-designated verifiers signatures. In Javier López, Sihan Qing, and Eiji Okamoto, editors, *ICICS 04*, volume 3269 of *LNCS*, pages 495–507. Springer, Berlin, Heidelberg, October 2004. doi:10.1007/978-3-540-30191-2_38.

[Ped92] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 129–140. Springer, Berlin, Heidelberg, August 1992. doi:10.1007/3-540-46766-1_9.

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *ASIACRYPT 2001, volume 2248 of LNCS. Springer, Berlin, Heidelberg*, pages 552–565, 2001. doi:10.1007/3-540-45682-1_32.

[SALY17] Shi-Feng Sun, Man Ho Au, Joseph K. Liu, and Tsz Hon Yuen. RingCT 2.0: A compact accumulator-based (linkable ring signature) protocol for blockchain cryptocurrency monero. In Simon N. Foley, Dieter Gollmann, and Einar Snekkenes, editors, *ESORICS 2017, Part II*, volume 10493 of *LNCS*, pages 456–474. Springer, Cham, September 2017. doi:10.1007/978-3-319-66399-9_25.

[SBWP03] Ron Steinfeld, Laurence Bull, Huaxiong Wang, and Josef Pieprzyk. Universal designated-verifier signatures. In Chi-Sung Laih, editor, *ASIACRYPT 2003*, volume 2894 of *LNCS*, pages 523–542. Springer, Berlin, Heidelberg, November / December 2003. doi:10.1007/978-3-540-40061-5_33.

[SKM04] Shahrokh Saeednia, Steve Kremer, and Olivier Markowitch. An efficient strong designated verifier signature scheme. In Jong In Lim and Dong Hoon Lee, editors, *ICISC 03*, volume 2971 of *LNCS*, pages 40–54. Springer, Berlin, Heidelberg, November 2004. doi:10.1007/978-3-540-24691-6_4.

[SWP04] Ron Steinfeld, Huaxiong Wang, and Josef Pieprzyk. Efficient extension of standard Schnorr/RSA signatures into universal designated-verifier signatures. In Feng Bao, Robert Deng, and Jianying Zhou, editors, *PKC 2004*, volume 2947 of *LNCS*, pages 86–100. Springer, Berlin, Heidelberg, March 2004. doi:10.1007/978-3-540-24632-9_7.

[SZM04] Willy Susilo, Fangguo Zhang, and Yi Mu. Identity-based strong designated verifier signature schemes. In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *ACISP 04*, volume 3108 of *LNCS*, pages 313–324. Springer, Berlin, Heidelberg, July 2004. doi:10.1007/978-3-540-27800-9_27.

[ZK02] Fangguo Zhang and Kwangjo Kim. ID-based blind signature and ring signature from pairings. In Yuliang Zheng, editor, *ASIACRYPT 2002*, volume 2501 of *LNCS*, pages 533–547. Springer, Berlin, Heidelberg, December 2002. doi:10.1007/3-540-36178-2_33.