Non-Interactive Composition of Sigma-Protocols via Share-then-Hash

Masayuki Abe[1], Miguel Ambrona[1], Andrej Bogdanov[2], Miyako Ohkubo[3], and Alon Rosen[4]

[1]NTT Secure Platform Laboratories
{masayuki.abe.cp, miguel.ambrona.fu}@hco.ntt.co.jp

[2]Chinese University of Hong Kong, andrejb@cse.cuhk.edu.hk

[3]Security Fundamentals Laboratory, CSR, NICT, m.ohkubo@nict.go.jp

[4]Herzliya Interdisciplinary Center, alon.rosen@idc.ac.il

Abstract. Proofs of partial knowledge demonstrate the possession of certain subsets of witnesses for a given collection of statements $x_1, \ldots, x_n$. Cramer, Damgård, and Schoenmakers (CDS), built proofs of partial knowledge, given "atomic" protocols for individual statements x_i, by having the prover randomly secret share the verifier's challenge and using the shares as challenges for the atomic protocols. This simple and highly-influential transformation has been used in numerous applications, ranging from anonymous credentials to ring signatures.
We consider what happens if, instead of using the shares directly as challenges, the prover first hashes them. We show that this elementary enhancement can result in significant benefits:

- the proof contains a *single* atomic transcript per statement x_i,
- it suffices that the atomic protocols are κ-special sound for $\kappa \geq 2$,
- when compiled to a signature scheme using the Fiat-Shamir heuristic, its unforgeability can be proved in the *non-programmable* random oracle model.

None of the above features is satisfied by the CDS transformation.

Keywords: sigma-protocols, random oracles, proof of partial knowledge

1 Introduction

The focus of this paper is three-move public-coin proof systems. In such protocols, a prover sends an initial message, a, to the verifier who answers back with a random challenge, e. The prover finally replies with z, based on which the verifier accepts or rejects the proof. Σ-protocols [**?**] are a special class of 3PC protocols that have been used as building blocks in a wide variety of applications, and have been the subject of intensive study.

One property that makes a Σ-protocol easy to work with is the so-called *2-special soundness*: given any pair of "colliding" transcripts, (a, e, z) and (a, e', z') for $e \neq e'$, one can efficiently extract a witness w for the instance x being proved.

The zero-knowledge property is exhibited using a specific type of simulator, which takes x and e as input, and outputs a and z that form an accepting transcript. Being public-coin, with a uniformly chosen challenge sent by the verifier, the protocol can be made non-interactive using the Fiat-Shamir heuristic [**?**], where the prover generates the challenge e on its own by applying a hash function modeled as a random oracle to the initial message a.

Several techniques for efficient composition of Σ-protocols can be found in the literature. Among them, the technique by Cramer, Damgård, and Schoenmakers (CDS for short) is the most popular and well-studied [**?**]. In its simplest form, the CDS technique is used for proving the *disjunction* of n statements $x_1, \ldots, x_n$, convincing the verifier that the prover knows a witness w for at least one of the statements x_i. To this end, the prover shares a given challenge e into challenges $e_1, \ldots, e_n$ under the constraint that $e = e_1 \oplus \cdots \oplus e_n$ and uses e_i as the challenge in an individual run of the Σ-protocol for statement x_i.

Since the prover can choose in advance all but one shared challenge e_{i^*} for which w_{i^*} is known, it may run the simulator on (x_i, e_i) for all $i \neq i^*$ and the prescribed prover algorithm on (x_{i^*}, w_{i^*}). This enables the prover to complete the protocol given a witness for at least one out of n instances. If the atomic protocols are 2-special sound, the compound protocol is 2-special sound as well.

The way in which the verifier challenge is secret-shared can be generalized to implement any composition predicate that is efficiently computable by a monotone span program [**?**]. Since the compound protocol remains a Σ-protocol, it can also be made non-interactive via the Fiat-Shamir heuristic. While security is proved in the random oracle model, it does not necessitate trusted setup which is often required by efficient non-interactive proofs.

1.1 Our Contribution

We propose a simple enhancement to the CDS composition method and show that it results in several desirable features. In simple terms, the modification can be described as follows:

"Hash each share before using it as a challenge."

As simple as it appears to be, this modification enjoys significant benefits over the original CDS transformation: (1) in computation and communication efficiency, (2) in allowing a wider variety of choices for the underlying atomic protocols, and (3) in the tightness of the analysis in the random oracle model. We now elaborate on each of these benefits separately.

Recycling of transcripts for repeated statements. In the CDS transformation, the transcript of the compound protocol contains one instance of the atomic protocol for each occurrence of a statement x_i in the formula or monotone span program. In contrast, our proposed transformation allows to "recycle" transcripts of atomic protocols and let them have a *single* appearance per x_i. This may result in savings in prover computation and communication, whenever base statements x_i occur

repeatedly, especially in cases where the monotone span program describing the compound statement cannot be simplified to have few occurrences of x_i.

Consider for example the following compound statement, described in disjunctive normal form: $(x_1 \wedge x_2) \vee (x_1 \wedge x_3) \vee (x_3 \wedge x_4)$. Notice that in this case the instance x_1 appears in two clauses (and so does x_3). When applying the CDS transformation, a prover (wishing to protect w_1 from leaking) must run independent executions of the atomic Σ-protocol for each appearance of x_1 in the formula. Otherwise, in case that the initial message a for proving x_1 is shared by two transcripts (a_1, e_1, z_1) and (a_1, e_1', z_1'), it may be the case that $e_1 \neq e_1'$ which would yield a colliding pair of transcripts, enabling, even an honest verifier, to extract the witness w_1 for x_1. In some cases one may be able to find an equivalent formula with fewer occurrences of specific variables. However, performing such simplifications in general is a non-trivial and potentially error-prone process. Furthermore, in some cases it may simply be not possible. Indeed, a recent implementation of compound statements [**?**] is aware of such issues and takes explicit care to refrain from merging the initial messages for the same statements in the formula. Their compiler halts when a repeated statement is detected and let the programmer decide what to do. Such issues were also explicitly considered in the original CDS protocol. When a share of a challenge exceeds the challenge space size, CDS explicitly require to repeat the atomic protocol for the same instance so that the joint challenge space covers the maximum length of the shared challenges.

By applying a hash function to the secret-shared challenges in all occurrences of x_i we compress and fit the challenge to the original challenge space size. Assuming that the hash function is modeled as a random oracle, soundness is guaranteed by the fact that hashed challenges are randomly and uniformly distributed. This allows us to run the atomic proof for a given instance x_i only once, independently of how many times it appears in the compound formula, hence simplifies the protocol. Furthermore, it improves both the running time of the prover and verifier, and reduces the size of the proof. Consider, for instance, the compound statement $(x_1 \wedge x_2) \vee (x_1 \wedge x_3) \vee (x_3 \wedge x_4)$ again. The CDS+FS combination would require six transcripts: one per literal. Ours leads to a proof with simply four transcripts: one per variable, regardless of the number of occurrences in the formula. More concretely, our proof consists of four transcripts $(a_1, e_1, z_1), ..., (a_4, e_4, z_4)$ together with secret shares (s_1, s_2, s_3, s_4) where each (a_i, e_i, z_i) is accepting with respect to the i-th Σ-protocol and $e_i = H(s_i)$. Furthermore, the shares are such that all qualified sets of shares (according to the *dual* access structure induced by formula) recover the secret $s := H(a_1, \ldots, a_4)$. In our example this could be enforced by setting $s_1 := \{d_1, d_2\}$, $s_2 := \{d_1\}$, $s_3 = \{d_2, d_3\}$, and $s_4 = \{d_3\}$ where $d_1 + d_2 + d_3 = s$. See Section 3.2 for a more detailed comparison between our scheme with previous work in terms of performance and proof size.

Wider choice for special soundness of atomic protocols. Special soundness is, by definition, restricted to the case where two colliding transcripts are necessary and sufficient for extracting a witness. However, some protocols in the literature are

only known to satisfy a more relaxed κ-special soundness requirement, in which $\kappa > 2$ colliding transcripts are necessary and sufficient for witness extraction.

The original CDS transformation was designed to only handle 2-special soundness, and indeed may totally lose soundness if applied to general κ-sound protocols for $\kappa > 2$ [**?**]. As an example of 3-special soundness, consider Stern's protocol [**?**], often used in the context of lattices and codes [**?**, **?**, **?**, **?**]. In its basic version, a challenge is chosen from $\{0, 1, 2\}$ and a cheating prover, or zero-knowledge simulator, having no witness can answer to two preliminary chosen challenge values out of the three. The original CDS technique for composing two runs of the protocol suggests to share challenge e as $e = e_1 + e_2 \bmod 3$ and use e_1 and e_2 as a challenge in each run. This is however totally insecure since a cheating prover may simulate on $e_1 \in \{0, 1\}$ and $e_2 \in \{1, 2\}$ and pick a proper combination of challenge values for e_1 and e_2 to fulfill the constraint $e = e_1 + e_2 \bmod 3$ for any challenge $e \in \{0, 1, 2\}$. Such an attack works even with parallel repetition of the protocol, with challenge space $\{0, 1, 2\}^\ell$ for polynomial ℓ, and even after applying the Fiat-Shamir transformation, as the adversary can similarly attack each coordinate individually and win with probability 1.

Applying an ideal hash function to e_1 and e_2 individually makes them uniformly distributed over the challenge space. With large enough challenge space, which can be obtained by parallel repetition of Stern's basic protocol, this virtually prevents a cheating prover from controlling the distribution of the challenges.

We prove that this intuition is valid in the random oracle model. As a result, our scheme is sound even for κ-special sound protocols with $\kappa > 2$. Other well-known examples of κ-special sound protocols ranges from the widely known GMW protocol for graph 3-colorability [**?**], and a useful protocol for a binary opening of Pedersen-like commitments (with 3-special soundness) [**?**], to a fun protocol for Sudoku puzzles [**?**].

Various flavors of soundness. We prove soundness in different flavors in the programmable and/or non-programmable random oracle models (NPROM) [**?**]. As shown in [**?**], when viewed as a non-interactive membership argument system, CDS composition with Fiat-Shamir (henceforth CDS+FS) is sound in NPROM provided that underlying protocols are optimally sound. Ours covers more relaxed statistically sound protocols.

If one of the two hash functions, one used for FS and the other used for hashing shares, is programmable random oracle, our construction provides simulation extractability [**?**], which is a strong form of knowledge soundness. If both are programmable, and the underlying protocol is unique response where z is unique for x, a, and e, it is strongly simulation extractable.

Unforgeability in non-programmable random oracle model. In a recent paper, Fischlin, Harasser, and Janson [**?**] show that when the CDS protocol is compiled into a signature scheme via the Fiat-Shamir transform, its unforgeability against adaptive chosen message attacks cannot be (black-box) proved in NPROM. They aregue that it contrasts to a *sequential* composition considered in [**?**].

The share-hashing in our construction circumvents this impossibility result. A key observation in the (im)possibility argument of [**?**] is that the sequential composition in [**?**] makes hash queries for each underlying protocol execution in some order, and the order of the queries reveals which instance the adversary is attacking. In contrast, CDS+FS makes a combined hash query for all underlying protocol executions at once, thus revealing no information which execution the adversary is attacking. This difference is precisely what renders the signature scheme via the sequential composition provably unforgeable in the NPROM, and CDS+FS not.

Since in our transformation hashing is applied for each execution of the underlying protocol, observing the order of the queries reveals which ones the adversary is attacking, just as in the example above. We are thus able to prove unforgeability in the NPROM, using the same proof strategy as developed in [**?**].

1.2 Applications

Our minor modification to the CDS+FS transformation means that it can serve as a plug-in replacement for most applications of the CDS protocol, with the only exceptions being the ones in which using a random oracle is not allowed.

In some cases the applicability of our transformation goes beyond what could have been achieved by CDS+FS. As a demonstration, consider a generic construction of a ring signature scheme [**?**] with the following added features: (1) it supports any monotone formula access structure, (2) it can be built from κ-special-sound Σ-protocols for hard languages, (3) it is unforgeable against chosen message and chosen ring attacks in the NPROM, and (4) it is setup-free in the sense that players do not need to interact to each other or to access public parameters (except for security parameter) to set up their public-keys.

The CDS+FS transformation is equipped with all the features mentioned above, and can be used to construct a secure signature scheme in a standard manner. However, we do not know how to prove its unforgeability in the NPROM, the main difficulty being that, unlike the case of a standard signature scheme, a ring-signature adversary is allowed to specify the access structure. Let us elaborate on this point further below.

In [**?**], it is shown that a non-interactive argument system for a simple cyclic graph representing a sequence of disjunctions can be turned into a secure signature scheme in the NPROM where the public key is a set of instances of a hard language. In the security argument, the reduction simulates signatures using a non-tight qualified set of instances, and, by observing queries to the random oracle, identifies which instance the adversary is attacking. It is then shown that replacing the target instance with an incorrect one that has no corresponding witness does not make much difference to the computationally limited adversary since those instances are supposed to be indistinguishable and signatures can still be simulated as the remaining correct instances form a qualified set.

In the attack scenario for ring signatures, however, it is the adversary who chooses the access structure. The adversary can ask a signature on a full set of

instances so that the only qualified set is tight. Accordingly, signatures cannot be simulated if an instance is turned into an incorrect one.

Our solution is to form each key by a disjunctive relation over two instances, and combine them into a single monotone formula. This allows to simulate signatures even if one of the pairs is turned into incorrect, and just as in [**?**] enables us to argue that attacking the incorrect instance is unsuccessful in the NPROM. The resulting scheme yields signatures whose size is linear in the number of involved public keys.

While there exist more compact ring signature schemes, e.g., [**?**], with logarithmic-size signatures and without using random oracles, our construction is more flexible in the choice of underlying building blocks and in the number of instantiations. This is on top of being the first scheme provable in the NPROM.

1.3 Related Work

Composition of proof systems. The task of proving compound statements in a zero-knowledge manner can be in principle realized generically by reducing to some NP-complete language, and in some cases even a flexible and convenient one such as satisfiability of Quadratic Arithmetic Programs. This approach is flexible, as it allows to dynamically adjust the statement to be proved depending on the application at hand. A popular application that has seen prominence recently is that of proving possession of a preimage of a value under a specified hash function. Recent implementations demonstrate reasonable performance, though we are still in early stage of deployment, and further progress is required.

Composition is an active topic also in the context of NIZKs in the common reference string model. There are number of existing techniques in the literature, e.g., [**?**, **?**, **?**, **?**, **?**, **?**, **?**], to implement disjunctive relations for the Groth-Sahai proofs [**?**] and Quasi-Adaptive NIZKs [**?**]. One of the common ideas is to use arithmetic relations of the form $x(x-1)=0$ that naturally translate to logical disjunctions: $(x=1) \vee (x=0)$. Another popular approach is to split a common reference string in two parts so that one of them can be used for simulation, whereas using a witness for the other part is unavoidable. In [**?**], Agrawal, Ganesh, and Mohassel studied efficient monotone composition of algebraic and non-algebraic statements combining both Σ-protocols and generic NIZKs for NP.

The composition technique most relevant to our work is that of ring-like sequential composition, introduced in [**?**] and revisited recently in [**?**], all of which admit soundness proofs in the NPROM. Recently, [**?**] consider a generalization of sequential composition to so-called *acyclicity programming* (a model that is closely related to branching programs), which in some cases goes beyond CDS composition, the latter being limited to monotone span programs in terms of expressibility. Still, generally speaking the two transformations are incomparable, and it should be mentioned that both CDS and our current transformation are able to easily handle the important case of threshold access structures. Precise proof sizes and computational costs are also incomparable as they depend on the structure of the compound relation.

Fiat-Shamir transform in NPROM and the standard model. The issue of programmability of random oracles in the case the Fiat-Shamir transform is discussed in [**?**, **?**]. They present an efficient FS transformation for constructing NIZK in the common reference and random oracle models whose zero-knowledge property does not rely on random oracles and only the proof of soundness requires a NPROM. The proof for soundness in the NPROM in [**?**] demands optimal soundness from the underlying protocol: for every false statement and every first message, there exists at most one challenge that has a valid response satisfying the verification predicate.

Not relying on programmable random oracles in the soundness argument of Fiat-Shamir transform may allow to instantiate the hash function under milder assumptions such as key dependent message secure encryption [**?**] or lattice-based assumptions [**?**, **?**] through the notion of correlation intractability [**?**]. They require the underlying protocol optimally sound [**?**] and design the hash function used in the FS transform so that it hardly outputs the bad challenge for which a valid response exists. Unfortunately, the additional hashing for generating challenges in our construction makes it hard to follow their approach as the bad challenge function will depend on the hash function.

Ring signatures. A fair number of papers devote themselves to improve and generalize the seminal work of ring signatures scheme in [**?**]. In [**?**], a general monotone access structure is supported for composition of signatures based on trapdoor permutations. A construction based on Σ-protocols is presented in [**?**] and extended in [**?**] with a simple mechanism for anonymity revocation, and in [**?**] with a support for threshold structures. These early works, followed by, e.g., [**?**], achieve the setup-free property in the *programmable* random oracle model. We note that the scheme in [**?**] hashes shared challenges to adjust the challenge size to incorporate RSA keys in a ring. When the ring consists only of the discrete-log type ones, it can be seen as a special case of our construction, a composition of Schnorr proofs with hashed shares, but none of the benefits claimed in this paper were considered.

There are number of schemes, e.g., [**?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**], that require trusted setup but provide more flexible access structures and/or achieves high performance when instantiated with mathematically rich primitives such as pairings, lattices, and codes. A scheme in [**?**] is favorable in that the security is proven in the standard model, no trusted setup is needed, and the proof size is logarithmic in the number of involved public-keys limiting the access structure only to a ring.

2 Preliminaries

For a finite set S, we write $a \leftarrow S$ to denote that a is uniformly sampled from S. We denote the security parameter by $\lambda \in \mathbb{N}$. Given two functions $f, g : \mathbb{N} \to [0, 1]$, we write $f \approx g$ if the difference $|f(\lambda) - g(\lambda)|$ is asymptotically smaller than the inverse of any polynomial. A function f is said to be *negligible* if $f \approx 0$, whereas

it is said to be *overwhelming* when $f \approx 1$. For integers m, n, such that $n \geq m$, we denote by $[m, n]$ the range $\{m, m+1, \ldots, n\}$. We denote by $[n]$ the range $[1, n]$. By $\mathbb{N}^*$ we denote the space of arbitrarily-long sequences of numbers in $\mathbb{N}$. When A is a probabilistic algorithm, we denote by $A(x; r)$ an execution of A on input x and random coin r taken from an appropriate domain defined for A. If the random coin is not important in the context, we simply write as $A(x)$.

Let $R : \mathcal{X} \times \mathcal{W} \to \{0, 1\}$ be a binary relation defined over a set of instances $\mathcal{X}$ and a set of witnesses $\mathcal{W}$. We write $(x, w) \in R$ as a shorthand for (x, w) satisfying $R(x, w) = 1$. For convenience, we separate instances according to the security parameter. By R_λ, we mean relation R on instances of length λ. Let L_R be the language defined as $L_R \coloneqq \{x \in \mathcal{X} \mid \exists w \in \mathcal{W} : R(x, w) = 1\}$. A statement is a relation on an instance, which is true if and only if the instance is in the language defined by the relation. We say that L_R is a hard language if $(x, w) \in R$ is efficiently and uniformly sampleable, and there exists $\tilde{L}$ that is efficiently sampleable, has no intersection with L_R, and is computationally indistinguishable from L_R. We abuse notation and write $(x, w) \leftarrow R$ to represent uniform sampling of (x, w) satisfying R. For a monotone access structure Γ over $[n]$ and a set of n relations $\boldsymbol{R} \coloneqq (R_1, \ldots, R_n)$, we denote by $\Gamma_{\boldsymbol{R}}$ a relation obtained by composing relation $R_i \in \boldsymbol{R}$ following structure Γ.

2.1 Σ-protocols

A Σ-protocol for relation R is a three-round public-coin proof system that is special honest verifier zero-knowledge and 2-special sound as defined in the following. It is witness indistinguishable and statistically sound. We also introduce additional security notions on which we rely when proving stronger properties about our construction.

Definition 1 (Three-round public-coin proof system). *A* three-round public-coin proof system *for relation R consists of algorithms $(\mathcal{C}, \mathcal{Z}, \mathcal{V})$ where:*

- $a \leftarrow \mathcal{C}(x, w\,; r)$ *computes an initial message, a, for the given instance x and witness w with a random coin r uniformly taken from an appropriate domain.*
- $z \leftarrow \mathcal{Z}(x, w, r, e)$ *computes an answer, z, for the given challenge $e \in \{0, 1\}^\mu$, and coin r used to generate a on x and w.*
- $1/0 \leftarrow \mathcal{V}(x, a, e, z)$ *outputs* 1 *or* 0 *for acceptance or rejection, respectively.*

We say a three-round public-coin proof system is complete *if for every $\lambda \geq 1$, every pair $(x, w) \in R$, where $|x| = \lambda$, for all $e \in \{0, 1\}^\mu$, for all $a \leftarrow \mathcal{C}(x, w\,; r)$, and for all $z \leftarrow \mathcal{Z}(x, w, r, e)$, $\mathcal{V}(x, a, e, z) = 1$ holds.*

Definition 2 (Special Honest Verifier Zero-Knowledge). *A three-round public-coin proof system $(\mathcal{C}, \mathcal{Z}, \mathcal{V})$ is* special honest verifier zero knowledge *if there exists a probabilistic polynomial-time algorithm $\mathcal{S}$ such that, for every stateful PPT adversary $\mathcal{A}$,*

$$\begin{aligned} &\Pr\left[(x, e) \leftarrow \mathcal{A}(1^\lambda);\ a \leftarrow \mathcal{C}(x, w; r);\ z \leftarrow \mathcal{Z}(x, w, r, e) : \mathcal{A}(a, z) = 1\right] \\ \approx\ &\Pr\left[(x, e) \leftarrow \mathcal{A}(1^\lambda);\ (a, z) \leftarrow \mathcal{S}(x, e) \qquad\qquad : \mathcal{A}(a, z) = 1\right] \end{aligned}$$

where r is sampled form the corresponding distribution and $\mathcal{A}$ must output values such that $(x, w) \in R$ and e is in $\{0,1\}^\mu$.

Definition 3 (Witness Inidistinguishability). *A three-round public-coin proof system $(\mathcal{C}, \mathcal{Z}, \mathcal{V})$ is* witness indistinguishable *if for all $x \in L_R$, and all w_1, w_2 satisfying $R(x, w_1) = R(x, w_2) = 1$, transcripts (a_1, e, z_1) and (a_2, e, z_2) distribute identically, where $a_i \leftarrow \mathcal{C}(x, w_i; r_i)$, $e \leftarrow \{0,1\}^\mu$, $z \leftarrow \mathcal{Z}(x, w_i, r_i, e)$ for $i = 1, 2$.*

Special soundness [**?**] is a special form of knowledge soundness which guarantees that, given two colliding transcripts $(x, a, \{e_1, z_1\}, \{e_2, z_2\})$, a witness w (for x) can be extracted efficiently if $e_1 \neq e_2$. A generalized form of this notion appears in the literature, e.g., [**?**, **?**, **?**, **?**]. Intuitively, κ-special soundness states that given κ-colliding transcripts $(x, a, \{e_1, z_1\}, \ldots, \{e_\kappa, z_\kappa\})$, a witness w can be extracted if all values $e_1, \ldots, e_\kappa$ are distinct. A question is from which distribution the challenges should be sampled and with how much probability the extraction should succeed. In some literature it is asked to hold for any e_i and to succeed perfectly. This is however too strong for our purpose as we would like to capture a wide variety of protocols, including the parallel version of Stern's protocol where an exponential number (but still negligible compared to the size of the challenge space) of colliding transcripts can be prepared without knowing the witness; on the other hand, a small number of collision over uniformly chosen challenges is sufficient for successful extraction with high probability. Consequently, we adopt the following definition.

Definition 4 (κ-Special Soundness). *A three-round public-coin proof system is* κ-special sound *with knowledge error ϵ if, there exists a deterministic polynomial-time algorithm $\mathcal{E}$ such that, for any stateful probabilistic polynomial-time adversary $\mathcal{A}$, and for all t polynomial in λ, it holds:*

$$\Pr\left[\begin{array}{c} (x,a) \leftarrow \mathcal{A}(1^\lambda) \\ e_1,\ldots,e_t \leftarrow \{0,1\}^\mu \\ (z_1,\ldots,z_t) \leftarrow \mathcal{A}(e_1,\ldots,e_t) \\ w \leftarrow \mathcal{E}(x,a,\{e_1,z_1\},\ldots,\{e_t,z_t\}) \end{array} : \begin{array}{c} \sum_{i=1}^t \mathcal{V}(x,a,e_i,z_i) \geq \kappa \\ \wedge \\ R(x,w) = 0 \end{array}\right] \leq \epsilon$$

where every e_i is distinct. It is special sound if ϵ is a negligible function and κ is polynomial in the security parameter. In particular, we say that it is perfectly special sound *if $\epsilon = 0$.*

There are different flavors of soundness as a proof of membership. An example is *optimal soundness*, which asserts that for any false instance x and any a, there exists at most one challenge e for which the transcript will pass the verification. In other words, for any $x \notin L_R$ and any a, and for all values $e \in \{0,1\}^\mu$ (except at most one), $\mathcal{V}(x, a, e, \cdot)$ is the zero function. We use more general statistical soundness allowing negligible error probability.

Definition 5 (Statistical Soundness). *A three-round public-coin proof system $(\mathcal{C}, \mathcal{Z}, \mathcal{V})$ is* statistically sound *with soundness error ϵ_{st} if for any (possibly unbounded) adversary $\mathcal{A}$, for all $x \notin L_R$ and all $a \in \{0,1\}^*$,*

$$\Pr[e \leftarrow \{0,1\}^\mu;\ z \leftarrow \mathcal{A}(x,a,e)\ :\ \mathcal{V}(x,a,e,z) = 1] < \epsilon_{\mathsf{st}}\ .$$

We say it is statistically sound if ϵ_{st} is negligible in λ.

In other words, a three-round public-coin proof system is statistical sound with bound ϵ_{st} if and only if for every $x \notin L_R$ and any $a \in \{0,1\}^*$, at most a ϵ_{st} fraction of challenges has an answer that passes the verification.

In order to achieve stronger variant of simulation soundness, we require the uniqueness of z for (x, a, e). This is the so-called *unique response* property [**?**, **?**] and was stated in [**?**] in a general form as follows.

Definition 6 (Quasi-unique response). *A Σ-protocol has* quasi-unique responses *if for any security parameter $\lambda \in \mathbb{N}$, any polynomial-size $\nu \in \{0,1\}^*$, and for any PPT algorithm, the probability that, given 1^λ and ν as input, the adversary outputs (x, a, e, z, z') satisfying $\mathcal{V}(x, a, e, z) = \mathcal{V}(x, a, e, z') = 1$ and $z \neq z'$ is negligible in λ.*

2.2 Non-Interactive Arguments

We define non-interactive argument systems in a way that captures Σ-protocols transformed by the Fiat-Shamir heuristics in the random oracle model. Let $\mathcal{R}$ be a random oracle that returns an independently and uniformly chosen value in an appropriate domain for every distinct input.

Definition 7 (Non-Interactive Argument System). *A* non-interactive argument system *for relation R in the random oracle model is a pair of polynomial-time oracle algorithms $(\mathsf{Prove}, \mathsf{Verify})$ that, for random oracle $\mathcal{R}$:*

- *$\pi \leftarrow \mathsf{Prove}^{\mathcal{R}}(x, w)$ is a probabilistic algorithm that takes an instance x and a witness w and outputs a proof π.*
- *$0/1 \leftarrow \mathsf{Verify}^{\mathcal{R}}(x, \pi)$ is a deterministic algorithm that takes x and π, and outputs either 1 or 0 representing acceptance or rejection, respectively.*

It is complete *if, for every sufficiently large $\lambda \in \mathbb{N}$, and every $(x, w) \in R$, $\mathsf{Verify}^{\mathcal{R}}(x, \mathsf{Prove}^{\mathcal{R}}(x, w))$ outputs 1 except with negligible probability in λ. The probability is taken over coins of Prove and $\mathcal{R}$.*

Definition 8 (Zero-Knowledge). *A non-interactive argument system $(\mathsf{Prove}, \mathsf{Verify})$ for relation R is zero-knowledge in the random oracle model if there exists a PPT stateful algorithm Sim that for all probabilistic polynomial-time distinguisher D, $\Pr[1 \leftarrow D^{\mathcal{R}, \mathcal{O}_1}(1^\lambda)] - \Pr[1 \leftarrow D^{\mathcal{O}_2}(1^\lambda)]$ is negligible in λ. $\mathcal{O}_1$ is an oracle that, given (x, w) as input, returns $\perp$ if $(x, w) \notin R$, else returns the output of $\mathsf{Prove}^{\mathcal{R}}(x, w)$. $\mathcal{O}_2$ and Sim have two input interfaces. $\mathcal{O}_2$ forwards any string given through the first interface to the first interface of Sim and returns its output. Given (x, w) as input to the second interface, $\mathcal{O}_2$ returns $\perp$ if $(x, w) \notin R$, else forwards x to the second interface of Sim and returns the output. The probability is taken over coins of D, $\mathcal{R}$, Prove, and Sim.*

Definition 9 (Soundness). *A non-interactive argument system $(\mathsf{Prove}, \mathsf{Verify})$ for L_R is* sound *if for any PPT oracle algorithm $\mathcal{A}$, any $x \notin L_R$, $\Pr[\pi \leftarrow \mathcal{A}^{\mathcal{R}}(x) : 1 = \mathsf{Verify}^{\mathcal{R}}(x, \pi)]$ is negligible in λ. The probability is taken over coins of $\mathcal{A}$ and $\mathcal{R}$.*

Simulation extractability is a stronger notion of simulation soundness. Intuitively, it guarantees that even after having seen simulated proofs on arbitrary instances, the adversary cannot create a valid proof on a fresh instance for which the knowledge extraction fails. This notion was defined in the common reference string model in [**?**] and in the random oracle model in [**?**].

Definition 10 (Simulation Extractability). *A non-interactive zero-knowledge argument system* $(\mathsf{Prove}, \mathsf{Verify})$ *for relation* R *with zero-knowledge simulator* Sim *is simulation extractable in the random oracle model if, for any PPT oracle algorithm* $\mathcal{A}$*, there exists an expected polynomial-time algorithm* $\mathcal{E}$ *for which the following experiment returns* 1*.*

$\mathsf{Expr}^{\mathsf{se}}_{\mathcal{A}}(\lambda)$:

1. *Run* $(x, \pi) \leftarrow \mathcal{A}^{\mathsf{Sim}}(1^{\lambda})$.
2. *Output* 1 *if* $0 \leftarrow \mathsf{Verify}^{\mathsf{Sim}}(x, \pi)$ *or* x *has been queried to the second interface of* Sim.
3. *Run* $w \leftarrow \mathcal{E}^{\mathcal{A}}(x, \pi, \sigma)$.
4. *Output* $b := R(x, w)$.

Parameter σ *is the view of* Sim*. It is strongly simulation extractable if the freshness condition in Step 2 is on* (x, π) *as a pair instead of just on* x*.*

The above definitions are for the programmable random oracle model. To cast non-programmable random oracles in the definitions, allow every entity direct access to the oracle [**?**].

3 The Share-then-Hash technique

3.1 Construction

Let n be a polynomial in λ. Let SS be a perfect secret sharing scheme over $\{0,1\}^{\mu}$ for an access structure over $[n]$ of size polynomial in n. Let Share be the sharing algorithm of SS, and $D(s)$ be distribution of outputs from $\mathsf{Share}(s)$. For qualified set A and secret $s \in \{0,1\}^{\mu}$, we denote by $D_A(s)$ the joint distribution of shares in A. We denote by A^{c} the set $[n] \backslash A$ and by $D_{A^{\mathsf{c}}}$ the distribution of shares for the non-qualified set A^{c} of A, which is independent of the secret (due to SS being a perfect secret sharing scheme). For a set of shares $S := (s_1, \ldots, s_n)$ and a set $A \subseteq [n]$, we denote by S_A the set of shares indexed by A, i.e., $S_A := \{s_i \in S \mid i \in A\}$. For the sake of readability, we assume that S_A identifies A from its data structure. A perfect secret sharing scheme over secret space $\{0,1\}^{\mu}$ for polynomial μ in λ is *semi-smooth* [**?**] if on top of standard polynomial-time and space requirements it satisfies the following properties:

- There exists a polynomial-time algorithm, $\mathsf{CheckShares}$ that, given a full set of shares and a secret, returns 1 if all qualified sets of shares recover the secret. It returns 0, otherwise.

- There exists a polynomial-time algorithm, $\mathsf{Complete}$ that, for any secret s, any non-qualified set A^{c}, and any set of shares $S_{A^{\mathsf{c}}} \in D_{A^{\mathsf{c}}}$, outputs a set of shares in $D(s)$ that includes $S_{A^{\mathsf{c}}}$ as shares for A^{c}.

Note that the presence of $\mathsf{CheckShares}$ does not imply that SS is a verifiable secret sharing scheme where, given a share s_i and public parameters, one can assure consistency of the share. Semi-smooth secret sharing schemes exist for threshold and general monotone access structures represented by monotone span programs [**?**].

Let Γ be a monotone access structure over $[n]$, and Γ^* be the *dual* of Γ defined as $A \in \Gamma^* \Leftrightarrow A^{\mathsf{c}} \notin \Gamma$ [**?**]. (Note that the dual operation is an involution, i.e., $(\Gamma^*)^* = \Gamma$.) Let $\mathsf{SS} = (\mathsf{Share}, \mathsf{CheckShares}, \mathsf{Complete})$ be a semi-smooth perfect secret sharing scheme over $\{0,1\}^\mu$ for Γ^*. Let $\boldsymbol{x} := (x_1, \ldots, x_n)$ be a set of instances and $\boldsymbol{w} := (w_1, \ldots, w_n)$ be a witness set where for a qualified set $A \in \Gamma$, let relation $R_i(x_i, w_i) = 1$ hold for all $i \in A$. Let $\Sigma_i = (\mathcal{C}_i, \mathcal{Z}_i, \mathcal{V}_i)$ be a sigma-protocol for relation R_i. We assume all Σ-protocols have a common challenge space $\{0,1\}^\mu$ for certain polynomial μ in security parameter λ. Let $H_e : \{0,1\}^* \to \{0,1\}^\mu$ and $H_c : \{0,1\}^* \to \{0,1\}^\mu$ be hash functions.

Theorem 1. *Figure 1 describes a non-interactive argument system for $\Gamma_{\boldsymbol{R}}$:*

- *It is complete and witness indistinguishable.*
- *It is zero-knowledge if H_c or H_e are modeled as programmable random oracles.*
- *It is a sound membership proof for language $L_{\Gamma_{\boldsymbol{R}}}$ if H_c and H_e are modeled as non-programmable random oracles and all Σ_i are statistically sound.*
- *It is simulation extractable if H_c and H_e are random oracles and at least one is programmable and if and all Σ_i are κ-special sound.*
- *It is strongly simulation extractable if both H_c and H_e are programmable random oracles, and all Σ_i are κ-special sound and unique response.*

Completeness and witness indistinguishability can be shown as in the original CDS+FS scheme. Zero-knowledge in the programmable random oracle model is assured by inspecting the simulators from Figure 2. The first simulator is for the case where H_c is programmable and the second one is for the case where H_e is programmable. In the following, we focus on soundness in different flavors and present a proof sketch for them, without stating concrete bounds, but our arguments are detailed enough to derive full proofs. We use the following proposition taken from [**?**].

Proposition 1. *Let Γ be monotone. A set is qualified in Γ if and only if it has a non-empty intersection with every qualified set in Γ^*.*

Proof (Of soundness as a membership proof system). Suppose that an adversary $\mathcal{A}$ outputs a valid proof $\hat{\pi} = \{(\hat{a}_i, \hat{z}_i), \hat{s}_i\}_{i \in [n]}$ on instance $\hat{\boldsymbol{x}} = (\hat{x}_1, \ldots, \hat{x}_n)$ and access structure Γ after making at most q queries to the random oracles. For the forged proof to be considered a valid forgery (in the soundness game), $\hat{\boldsymbol{x}}$ must be a false instance (with respect to Γ), i.e., for every qualified set $A \in \Gamma$, there must exist some $i \in A$ such that $x_i \notin L_{R_i}$. Furthermore, $\mathsf{CheckShares}_{\Gamma^*}(\hat{s}, \hat{s}_1, \ldots, \hat{s}_n)$

$\mathsf{Prove}_{\Gamma}(\boldsymbol{x}, \boldsymbol{w})$:

1. Set $A := \{\, i \,|\, R_i(x_i, w_i) = 1 \,\forall i \in [n]\}$ and $A^{\mathsf{c}} := [n] \setminus A$.
2. Sample $s' \leftarrow \{0,1\}^{\mu}$, and set $(s'_1, \ldots, s'_n) \leftarrow \mathsf{Share}_{\Gamma^*}(s')$, $S_{A^{\mathsf{c}}} := \{s'_i \,|\, i \in A^{\mathsf{c}}\}$.
3. For all $i \in A^{\mathsf{c}}$, set $e_i := H_e(\Gamma, \boldsymbol{x}, i, s'_i)$, and run $(z_i,\ a_i) \leftarrow \mathcal{S}_i(x_i, e_i)$.
4. For all $i \in A$, set $a_i \leftarrow \mathcal{C}_i(x_i, w_i; r_i)$.
5. Set $s := H_c(\Gamma, \boldsymbol{x}, a_1, \ldots, a_n)$, and $(s_1, \ldots, s_n) \leftarrow \mathsf{Complete}_{\Gamma^*}(s, S_{A^{\mathsf{c}}})$.
6. For all $i \in A$, set $e_i := H_e(\Gamma, \boldsymbol{x}, i, s_i)$, and run $z_i \leftarrow \mathcal{Z}_i(x_i, w_i, r_i, e_i)$.
7. Return $\pi := \{(a_i, z_i), s_i\}_{i \in [n]}$.

$\mathsf{Verify}_{\Gamma}(\boldsymbol{x}, \pi)$:

1. Parse π as $\{(a_i, z_i), s_i\}_{i \in [n]}$.
2. Set $s := H_c(\Gamma, \boldsymbol{x}, a_1, \ldots, a_n)$ and $e_i := H_e(\Gamma, \boldsymbol{x}, i, s_i)$ $\forall i \in [n]$.
3. Return $\bigwedge_{i \in [n]} \mathcal{V}_i(x_i, a_i, e_i, z_i) \ \wedge\ \mathsf{CheckShares}_{\Gamma^*}(s, s_1, \ldots, s_n)$.

Fig. 1: Share-then-Hash CDS+FS for relation $\Gamma_{\boldsymbol{R}}$.

must be 1, for $\hat{s} := H_c(\Gamma, \hat{\boldsymbol{x}}, \hat{a}_1, \ldots, \hat{a}_n)$; and $\mathcal{V}_i(\hat{x}_i, \hat{a}_i, \hat{e}_i, \hat{z}_i)$ must be accepting for $\hat{e}_i := H_e(\Gamma, \hat{\boldsymbol{x}}, i, \hat{s}_i)$ and all $i \in [n]$.

If for some $i^* \in [n]$ such that $x_{i^*} \notin L_{R_{i^*}}$ the adversary did not make query $H_e(\Gamma, \hat{\boldsymbol{x}}, i^*, \hat{s}_i)$, since value $\hat{e}_{i^*}$ is assigned uniformly at random by H_e, the probability that $1 = \mathcal{V}_{i^*}(x_{i^*}, a_{i^*}, e_{i^*}, z_{i^*})$ for already fixed x_{i^*}, a_{i^*}, and z_{i^*} is at most $\epsilon_{\mathsf{st}} := \max_{i \in [n]}(\epsilon_{\mathsf{st}_i})$ where ϵ_{st_i} is the statistical soundness error of Σ_i. Similarly, if $H_c(\hat{\boldsymbol{x}}, \hat{a}_1, \ldots, \hat{a}_n)$ was not queried by the adversary, after the random assignemt of $\hat{s}$, by H_c, the probability that $\mathsf{CheckShares}_{\Gamma^*}(\hat{s}, \hat{s}_1, \ldots, \hat{s}_n)$ is successful is at most $2^{-\mu}$ ($\hat{s}$ must be equal to the value determined by $\hat{s}_1, \ldots, \hat{s}_n$).

Now, let Ω be the set of indices $i \in [n]$ where $x_i \notin L_{R_i}$ holds and $\hat{e}_i := H_e(\Gamma, \hat{\boldsymbol{x}}, i, \hat{s}_i)$ appears before $\hat{s} := H_c(\Gamma, \hat{\boldsymbol{x}}, \hat{a}_1, \ldots, \hat{a}_n)$ in the view of $\mathcal{A}$. First, assume that for all qualified sets $A \in \Gamma$, $A \cap \Omega$ is not empty. In virtue of Proposition 1, Ω must be a qualified set in Γ^* and thus, $\{\hat{s}_i\}_{i \in \Omega}$ uniquely determines a secret, s^*. Therefore, $\mathsf{CheckShares}_{\Gamma^*}(\hat{s}, \hat{s}_1, \ldots, \hat{s}_n) = 1$ will be satisfied only if $\hat{s}$ equals s^*, which happens with probability at most $2^{-\mu}$ since $\hat{s}$ is randomly assigned by H_c independently of $\{\hat{s}_i\}_{i \in \Omega}$.

Finally, suppose that there exists $A \in \Gamma$ with $A \cap \Omega = \emptyset$. In this case, there must exist $i^* \in A$ with $x_{i^*} \notin L_{R_{i^*}}$ (remember that $\hat{\boldsymbol{x}}$ is a false instance) and such that query $\hat{e}_{i^*} := H_e(\Gamma, \hat{\boldsymbol{x}}, i^*, \hat{s}_{i^*})$ appears after query $\hat{s} := H_c(\Gamma, \hat{\boldsymbol{x}}, \hat{a}_1, \ldots, \hat{a}_n)$ in the view of $\mathcal{A}$. Then, the probability that there exists a $\hat{z}_{i^*}$ that can satisfy $\mathcal{V}_{i^*}(\hat{x}_{i^*}, \hat{a}_{i^*}, \hat{e}_{i^*}, \hat{z}_{i^*}) = 1$ for fixed $(\hat{x}_{i^*}, \hat{a}_{i^*})$ is upper-bound by the statistical soundness error of Σ_{i^*}, which is upper-bounded by ϵ_{st}.

Accordingly, a valid proof on a false statement can be produced with probability at most $2\epsilon_{\mathsf{st}} + 2^{-\mu}$. □

$\mathsf{Sim1}_{\Gamma}(\boldsymbol{x})$:

1. Sample $s \leftarrow \{0,1\}^{\mu}$, and set $(s_1, \ldots, s_n) \leftarrow \mathsf{Share}_{\Gamma^*}(s)$.
2. For all $i \in [n]$, set $e_i := H_e(\Gamma, \boldsymbol{x}, i, s_i)$, and $(z_i, a_i) \leftarrow \mathcal{S}_i(x_i, e_i)$.
3. Program H_c to output s on input $(\Gamma, \boldsymbol{x}, a_1, \ldots, a_n)$.
4. Return $\pi := \{(a_i, z_i), s_i\}_{i \in [n]}$.

$\mathsf{Sim2}_{\Gamma}(\boldsymbol{x})$:

1. For all $i \in [n]$, set $e_i \leftarrow \{0,1\}^{\mu}$, and $(z_i, a_i) \leftarrow \mathcal{S}_i(x_i, e_i)$.
2. Set $s := H_c(\Gamma, \boldsymbol{x}, a_1, \ldots, a_n)$, and $(s_1, \ldots, s_n) \leftarrow \mathsf{Share}_{\Gamma^*}(s)$.
3. For all $i \in [n]$, program H_e to output e_i on input $(\Gamma, \boldsymbol{x}, i, s_i)$.
4. Return $\pi := \{(a_i, z_i), s_i\}_{i \in [n]}$.

Fig. 2: Zero-knowledge simulators.

Proof (Of simulation extractability). We first prove the case where H_c is programmable and H_e is non-programmable. Suppose that adversary $\mathcal{A}$ playing in the simulation extractability game, running in time t and performing at most q queries to the random oracle, outputs an instance $\hat{\boldsymbol{x}} = (\hat{x}_1, \ldots, \hat{x}_n)$ and a valid proof $\hat{\pi} = \{(\hat{a}_i, \hat{z}_i), \hat{s}_i\}_{i \in [n]}$ on $\hat{\boldsymbol{x}}$ with probability δ. For the output to be valid, it must hold that $\mathsf{CheckShares}_{\Gamma^*}(\hat{s}, \hat{s}_1, \ldots, \hat{s}_n) = 1$ for $\hat{s} := H_c(\Gamma, \hat{\boldsymbol{x}}, \hat{a}_1, \ldots, \hat{a}_n)$ and, additionally, for all $i \in [n]$, $\mathcal{V}_i(\hat{x}_i, \hat{a}_i, \hat{e}_i, \hat{z}_i) = 1$, where $\hat{e}_i := H_e(\Gamma, \hat{\boldsymbol{x}}, i, \hat{s}_i)$. Furthermore, $\hat{\boldsymbol{x}}$ must be different from any instance $\boldsymbol{x}$ observed by the simulation oracle.

The extractor runs the code of $\mathcal{A}$, simulating the proving oracle using $\mathsf{Sim1}$ in Figure 2 until a valid proof $\hat{\pi} = \{(\hat{a}_i, \hat{z}_i), \hat{s}_i\}_{i \in [n]}$ on an instance $\boldsymbol{x}$ is produced. The extractor then identifies the query $H_c(\Gamma, \hat{\boldsymbol{x}}, \hat{a}_1, \ldots, \hat{a}_n)$ in the adversaries execution and forks the execution at this point by providing a different uniformly chosen value as an answer to this query. By repeating the above forking $2\tau/\delta$ times for $\tau := \kappa n$, the extractor obtains τ valid proofs with a constant probability. We now argue that, if τ random secrets $\hat{s}^{(i)}$ for $i = 1, \ldots, \tau$ are shared to n players in a way that they pass $\mathsf{CheckShares}$ consistency check, then, for every qualified set of players, there is a player who receives at least κ distinct shares. The following lemma states it formally.

Lemma 1. *For sufficiently large polynomial μ in λ, for any semi-smooth secret sharing scheme over $\{0,1\}^{\mu}$, for any small constant κ, for any constant $\tau \geq \kappa n - 2n + 2$, for any stateless unbound algorithm $\mathcal{B}$, the following experiment returns 1 with negligible probability in λ.*

1. *For $i = 1$ to τ, do $s^{(i)} \leftarrow \{0,1\}^{\mu}$, and $(s_1^{(i)}, \ldots, s_n^{(i)}) \leftarrow \mathcal{B}(s^{(i)})$.*

2. *Return* 1 *if* $1 = \mathsf{CheckShares}_{\Gamma^*}(s^{(i)}, s_1^{(i)}, \ldots, s_n^{(i)})$ *for all* $i = 1, \ldots, \tau$ *and there exists a qualified set,* A*, such that, for each* $j \in A$*, number of distinct shares among* $s_j^{(1)}, \ldots, s_j^{(\tau)}$ *is less than* κ*. Return* 0*, otherwise.*

We first prove the following claim.

Claim 1. Let A be a qualified set and assume $\tau \geq \kappa|A| - 2|A| + 2$. The probability that for all $j \in A$, the set $S_j^{(\tau)} = \{s_j^{(1)}, \ldots, s_j^{(\tau)}\}$ has size less than κ is at most $(\tau - 1)(\kappa - 1)^{|A|}2^{-\mu}$.

Proof (Of Lemma 1). Set $\tau = \kappa n - 2n + 2$. By Claim 1 and a union bound, the probability that there exists a qualified set A such that $|S_j^{(\tau)}| < \kappa$ for all $j \in A$ is at most $2^n \cdot (\tau - 1)(\kappa - 1)^n 2^{-\mu}$. If this is not the case, then the set A of all j such that $|S_j^{(\tau)}| < \kappa$ is not qualified. □

Proof (Of Claim 1). We will show that as long as all sets $S_j^{(i)}$, $j \in A$ have size less than κ, the probability that $\sum_{j \in A} |S_j^{(i+1)}| = \sum_{j \in A} |S_j^{(i)}|$ is at most $(\kappa - 1)^{|A|}2^{-\mu}$. Initially, $\sum_{j \in A} |S_j^{(1)}| = |A|$. By a union bound over $1 \leq i < \tau$, $\sum_{j \in A} |S_j^{(\tau)}| \geq |A| + \tau - 1$ with probability at least $1 - (\tau - 1)(\kappa - 1)^{|A|}2^{-\mu}$. By our choice of τ, this condition implies $|S_j^{(\tau)}| \geq \kappa$ for some $j \in A$. By the reconstruction property, there is an *injective* function R_A that maps valid sequences $(s_j : j \in A)$ of shares to secrets $s \in \{0,1\}^\mu$. Assuming $|S_j^{(i)}| < \kappa$ for all $j \in A$, the image of R_A evaluated on the product set $\prod_{j \in A} S_j^{(i)}$ can have size at most $(\kappa - 1)^{|A|}$. So if $s^{(i+1)}$ is chosen at random from $\{0,1\}^\mu$, then the probability it belongs to the image of $R_A(\prod_{j \in A} S_j^{(i)})$ is at most $(\kappa - 1)^{|A|}2^{-\mu}$. By the injectivity of R_A, for any sequence $(s_j^{(i+1)} : j \in A)$ that reconstructs to $s^{(i+1)}$, $s_j^{(i+1)}$ must reside outside $S_j^{(i)}$ for at least one party $j \in A$, so the sum $\sum_{j \in A} |S_j^{(i)}|$ grows as desired. □

According to Lemma 1, with non-negligible probability, it holds that, for every qualified set $A \in \Gamma^*$, there exists $i \in A$ that yields $(\hat{a}_i, (\hat{s}_i^{(1)}, \hat{z}_i^{(1)}), \ldots, (\hat{s}_i^{(\kappa)}, \hat{z}_i^{(\kappa)}))$ that satisfies $1 = \mathcal{V}_i(\hat{x}_i, \hat{a}_i, \hat{e}_i^{(j)}, \hat{z}_i^{(j)})$ for $\hat{e}_i^{(j)} := H_e(\Gamma, \hat{\boldsymbol{x}}, i, \hat{s}_i^{(j)})$. Since all $\hat{e}_i^{(j)}$ are distinct except for negligible probability due to the uniform output from H_e, we have κ-colliding transcript $(\hat{a}_i, (\hat{e}_i^{(1)}, \hat{z}_i^{(1)}), \ldots, (\hat{e}_i^{(\kappa)}, \hat{z}_i^{(\kappa)}))$ over uniformly chosen challenges, which allows to extract $\hat{w}_i$ with overwhelming probability. What remains is the same as the knowledge soundness proof of the original CDS scheme; according to Proposition 1, there exists a qualified set A in Γ for which $\hat{w}_i$ for all $i \in A$ are extracted.

We next sketch a proof for the case where H_c is non-programmable and H_e is programmable. This time we do not require Lemma 1. The extractor first runs the adversary until it obtains a valid forgery. Proof queries from the adversary is answered by executing $\mathsf{Sim2}$ in Figure 2, which programs at most n random points on H_e in each invocation. Then the extractor rewinds the adversary to the point where it first receives $\hat{s}$ for query $H_c(\Gamma, \hat{\boldsymbol{x}}, \hat{a}_1, \ldots, \hat{a}_n)$. The extractor then continues the simulation as well as the first run except that it answers every fresh query to H_e with an independently chosen random value. These queries

to H_e made after receiving $\hat{s}$ from H_c are for a qualified set, $A \in \Gamma$, as we observed in the proof of soundness since otherwise $\mathsf{CheckShares}_{\Gamma^*}(\hat{s}, \hat{s}_1, \ldots, \hat{s}_n)$ in the verification returns 1 with probability at most $2^{-\mu}$. By repeating the above rewinding $2\kappa/\delta$ times, the extractor obtains κ valid forged proofs on $(\boldsymbol{x}, \hat{a}_1, \ldots, \hat{a}_n)$ with a constant probability. The forged proofs constitute κ colliding transcripts for each $x_{i \in A}$ unless random assignments to H_e collide by chance. Thus, by running the κ-special soundness extractor with the colliding transcripts as an input, a valid witness is obtained except for a negligible probability. We finally note that H_c must still be modeled as (non-programmable) random oracle to assure that $\hat{a}_i$ is fixed before $\hat{s}_i$ is queried to H_c. □

Proof (Of strong simulation extractability). This time, we relax the condition on $(\hat{\boldsymbol{x}}, \hat{\pi})$ so that it must be different from any pair $(\boldsymbol{x}, \pi)$ observed by the simulation oracle. As we have already proved the case of $\hat{\boldsymbol{x}} \neq \boldsymbol{x}$ in the above, we consider $\hat{\boldsymbol{x}} = \boldsymbol{x}$ and $\hat{\pi} \neq \pi$ happens for some $(\boldsymbol{x}, \pi)$ observed by the simulation oracle. Let $\pi = \{(a_i, z_i), s_i\}_{i \in [n]}$. If $(\hat{a}_1, \ldots, \hat{a}_n) \neq (a_1, \ldots, a_n)$, then we fork at query $H_c(\Gamma, \hat{\boldsymbol{x}}, \hat{a}_1, \ldots, \hat{a}_n)$ and do the same as done in the proof of simulation extractability. Otherwise, if $(\hat{a}_1, \ldots, \hat{a}_n) = (a_1, \ldots, a_n)$ and $(\hat{s}_1, \ldots, \hat{s}_n) \neq (s_1, \ldots, s_n)$, we again fork at query $H_c(\Gamma, \hat{\boldsymbol{x}}, \hat{a}_1, \ldots, \hat{a}_n)$. Observe that the query is made by zero-knowledge simulator. So we cannot answer to the newly assigned value with the same $\hat{a}_i$. We instead simulate by using the same (a_i, e_i, z_i) for every $i \in [n]$. It can be done by programming H_e with the same output $\hat{e}_i$ on a new input s_i in each fork. More precisely, for every new assignment of $s^{(j)}$ to $H_c(\Gamma, \hat{\boldsymbol{x}}, \hat{a}_1, \ldots, \hat{a}_n)$ in the j-th fork, compute $(s_1^{(j)}, \ldots, s_n^{(j)}) \leftarrow \mathsf{Share}_{\Gamma^*}(s^{(j)})$. Then define $H_e(\Gamma, \boldsymbol{x}, i, s_i^{(j)})$ by e_i used in the original run and answer with the same z_i. Accordingly, though shares s_i appear in the respective challenge round differ in every fork, simulated transcript (a_i, e_i, z_i) remains the same. Now, τ successful forks leads to extracting witness in a qualified set in Γ as before. Due to the quasi-unique response property, we are already done since $(\hat{x}_i, \hat{a}_i, \hat{s}_i) = (x_i^{(k)}, a_i^{(k)}, s_i^{(k)})$ cannot accommodate with restriction $(\hat{x}_i, \hat{a}_i, \hat{s}_i, \hat{z}_i) \neq (x_i^{(k)}, a_i^{(k)}, s_i^{(k)}, z_i^{(k)})$ except for negligible probability. □

3.2 Comparison with CDS

In order to illustrate the efficiency gain and the recycling technique of our new construction, consider the following general DNF formula on n-variables:

$$f(x_1, \ldots, x_n) = (x_{j_{\{1,1\}}} \wedge \ldots \wedge x_{j_{\{1,m_1\}}}) \vee \ldots \vee (x_{j_{\{\ell,1\}}} \wedge \ldots \wedge x_{j_{\{\ell,m_\ell\}}}) \ , \qquad (1)$$

and let $N \coloneqq \sum_{i=1}^{\ell} m_i$ be the total number of literals in f. Let Γ be the access structure over $[n]$ induced by f, and consider the following well-known and widely used perfect secret sharing of $s \in \mathbb{Z}_p$ (for some μ-bits prime p) under policy Γ^*:

$\mathsf{Share}_{\Gamma^*}(s)$:

sample $d_1, \ldots, d_\ell \leftarrow \mathbb{Z}_p$ uniformly restricted to $d_1 + \ldots + d_\ell = s$;
set $s_i \coloneqq \left\{ d_k \,|\, i \in \{j_{\{k,1\}}, \ldots, j_{\{k,m_k\}}\} \, \forall k \in [\ell] \right\} \ \ \forall i \in [n]$;
return $(s_1, \ldots, s_n)$.

Property	Proof system	
	CDS+FS	Share-then-Hash CDS+FS
Proof size	$N(\alpha+\zeta)+\mu\ell$	$n(\alpha+\zeta)+\mu\ell$
Optimized proof size[†]	$N\zeta+\mu\ell$	$n\zeta+\mu\ell$
Support for $(\kappa>2)$-special soundness	✗	✓
Unforgeability in NPROM[‡]	✗	✓

[†] When every a is uniquely identified and efficiently recoverable given (e,z).
[‡] When considered as a signature scheme. See Section 4.

Table 1: Comparison between previous work ([**?**, **?**]) and the Share-then-Hash CDS (this work). Values N, n and ℓ represent the number of literals, number of variables and number of clauses in the DNF formula (1) respectively. Value α (respectively ζ) represents the size in bits of the *first message* (respectively *last message*) of sigma protocols Σ_i. (Challenges are assumed to belong in $\{0,1\}^\mu$.)

The CDS+FS technique would yield a proof for $\Gamma_{\boldsymbol{R}}$ consisting of N transcripts where for all $k\in[\ell]$ and $k'\in[m_k]$, transcript $(a_{\{k,k'\}},e_k,z_{\{k,k'\}})$ is accepting with respect to the $j_{\{k,k'\}}$-th Σ-protocol. Also, for $s\coloneqq H(\Gamma,\boldsymbol{x},a_{\{1,1\}},\ldots,a_{\{\ell,m_\ell\}})$, it must hold $e_1+\cdots+e_\ell=s$. This results in a total proof size in bits of:

$$\mu\ell+\textstyle\sum_{k\in[\ell]}\left(\sum_{k'\in[m_k]}|a_{\{k,k'\}}|+|z_{\{k,k'\}}|\right)\ .$$

Instead, with our scheme from Figure 1, the resulting proof consists of n transcripts $\{(a_i,z_i)\}_{i\in[n]}$ together with a set of shares $\{s_i\}_{i\in[n]}$ produced by the above Share algorithm. Transcript (a_i,e_i,z_i) is accepting with respect the i-th Σ-protocol, where $e_i\coloneqq H(\Gamma,\boldsymbol{x},i,s_i)$ for every $i\in[n]$ and for $s\coloneqq H(\Gamma,\boldsymbol{x},a_1,\ldots,a_n)$, $\mathsf{CheckShares}(s,s_1,\ldots,s_n)=1$. In this case, the total proof size in bits results in:

$$|(s_1,\ldots,s_n)|+\textstyle\sum_{i\in[n]}|a_i|+|z_i| \ \ =^{1}\ \ \mu\ell+\sum_{i\in[n]}|a_i|+|z_i|$$

We refer to Table 1 for a more detailed comparison between the two proof systems. For simplicity, we assume that all Σ-protocols require first messages of similar length say $|a|=\alpha$, and also last messages of similar length $|z|=\zeta$. Some Σ-protocols are such that, given (e,z), there exists a unique value of a that makes the transcript accepting and that can be efficiently computed. In those cases, it is possible to optimize the proof size by not including the a value of any transcript. During verification, the omitted values are computed from the corresponding (e,z). Notice that this optimization can be applied to both schemes and it does not compromise soundness, since the prover has committed to the final share s (dependent of the a values) through the random oracle H.

[1] Although the total length of secrets $(s_1,\ldots,s_n)$ is μN, as above it is enough to store the ℓ disjunction values $(d_1,\ldots,d_\ell)$ sampled by Share.

Further optimizations may be possible, e.g. reducing the number of shares that appear in the proof, depending on the access structure.

Observe that the size of proofs produced with the share-then-hash technique can be dramatically smaller than the size of proofs with standard CDS+FS since, in general, N can be much larger than n. This improvement comes from the fact that share-then-hash proofs include exactly 1 transcript per atomic statement, which is a notable improvement since many practical scenarios involve complex and heavy sigma protocols. Having to produce (and then verify) independent transcripts for the same statement would be undesirable. Finally, notice that this optimization also brings computational savings since fewer transcripts need to be produced.

4 Application

This section presents a general ring signature scheme that supports monotone structures and is unforgeable against chosen message and chosen ring attacks in the NPROM. Note that when $n = 1$ the syntax and unforgeability of ring signature schemes reduce to those for ordinary signature schemes.

Definition 11 (General Ring Signature Scheme). *A ring signature scheme* RS *is triple of polynomial-time algorithm, described by* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *such that*

- $\mathsf{KeyGen}(1^\lambda)$: *It takes an input the security parameter* 1^λ *and outputs a pair* (vk, sk) *of verification and signing key. This execution is proceeded individually by each player.*
- $\mathsf{Sign}(\boldsymbol{vk}, \boldsymbol{sk}, msg, \Gamma)$: *It takes a set of verification keys* $\boldsymbol{vk} := (vk_1, \ldots, vk_n)$*, a monotone access structure* Γ *over* $[n]$*, a set of secret keys* $\boldsymbol{sk}$*, and a message* $msg \in \{0,1\}^*$ *and outputs a signature* σ*.*
- $\mathsf{Verify}(\boldsymbol{vk}, msg, \sigma, \Gamma)$: *It takes* $\boldsymbol{vk}, msg, \sigma$*, and* Γ*, and outputs either* 1 *for acceptance, or* 0 *for rejection.*

It is correct, if, for every $\lambda \in \mathbb{N}$*,* $n \geq 1$*, any monotone access structure* Γ *over* $[n]$*, any* $\boldsymbol{vk} := (vk_1, \ldots, vk_n)$ *and* $\boldsymbol{sk} := (sk_1, \ldots, sk_n)$ *that there exists* $A \in \Gamma$ *such that* $(vk_i, sk_i) \in \mathsf{KeyGen}(1^\lambda)$ *holds for all* $i \in A$*, for all* $msg \in \{0,1\}^*$*,* $\mathsf{RS.Verify}(\boldsymbol{vk}, msg, \mathsf{Sign}(\boldsymbol{vk}, \boldsymbol{sk}, msg, \Gamma), \Gamma) = 1$ *holds except for negligible probability.*

Definition 12 (Signer Anonymity). *A ring signature scheme is anonymous if, for any* $\lambda \in \mathbb{N}$*, any* $n \geq 1$*, any monotone structure* Γ *over* $[n]$*, any* $\boldsymbol{vk} = (vk_1, \cdots, vk_n)$*, and any* $\boldsymbol{sk}^{(b)} := (sk_1^{(b)}, \ldots, sk_n^{(b)})$ *for* $b = 0, 1$ *that there exists* $A \in \Gamma$ *such that* $(vk_i, sk_i^{(b)}) \in \mathsf{KeyGen}(1^\lambda)$ *holds for all* $i \in A$*, and for any* $msg \in \{0,1\}^*$*, two distributions* $(\boldsymbol{vk}, msg, \mathsf{Sign}(\boldsymbol{vk}, \boldsymbol{sk}^{(0)}, msg, \Gamma), \Gamma)$ *and* $(\boldsymbol{vk}, msg, \mathsf{Sign}(\boldsymbol{vk}, \boldsymbol{sk}^{(1)}, msg, \Gamma), \Gamma)$ *are statistically indistinguishable.*

Definition 13 (Unforgeability). *A ring signature scheme is unforgeable against adaptive chosen message and chosen ring attacks if for any sufficiently large λ, any $n \geq 1$, any polynomial-time adversary $\mathcal{A}$, the following experiment returns 1 only with negligible probability in λ.*

$\mathsf{Expr}^{\mathsf{euf}}_{\mathsf{RS},\mathcal{A}}(\lambda)$:

1. *Run $(vk_i, sk_i) \leftarrow \mathsf{RS.KeyGen}(1^\lambda)$ for $i \in [n]$. Initialize U with $\emptyset$.*
2. *Run $(\hat{\boldsymbol{vk}}, \hat{msg}, \hat{\pi}, \hat{\Gamma}) \leftarrow \mathcal{A}^{\mathcal{S},\mathcal{C}}(\boldsymbol{vk})$ where $\mathcal{S}$ and $\mathcal{C}$ are oracles that:*
   - *$\mathcal{S}$: Given $(\boldsymbol{vk}', msg, \Gamma, A)$ as input, if $\boldsymbol{vk}' \subseteq \boldsymbol{vk}$, Γ is a monotone structure over $[n'] := [|\boldsymbol{vk}'|]$, and $A \in \Gamma$, it returns $\sigma \leftarrow \mathsf{RS.Sign}(\boldsymbol{vk}', \boldsymbol{sk}', msg, \Gamma)$ where $\boldsymbol{sk}' = (sk_1, \ldots, sk_{n'})$ that $(vk_i, sk_i) \in \mathsf{RS.KeyGen}(1^\lambda)$ for all $i \in A$ and $sk_i = \perp$ for all $i \in [n'] \setminus A$. It returns $\perp$, otherwise.*
   - *$\mathcal{C}$: Given $i \in [n]$, it adds vk_i to U, and returns sk_i.*
3. *Output 1 if all the following conditions are met.*
   - $1 = \mathsf{RS.Verify}(\hat{\boldsymbol{vk}}, \hat{msg}, \hat{\pi}, \hat{\Gamma})$
   - $\hat{\boldsymbol{vk}} \subseteq \boldsymbol{vk}$
   - $\forall A \in \hat{\Gamma}$, $\{\hat{vk}_i \in \hat{\boldsymbol{vk}} \mid i \in A\} \not\subseteq U$
   - *$(\hat{\boldsymbol{vk}}, \hat{msg}, \hat{\Gamma})$ has never been submitted to $\mathcal{S}$*

   Otherwise output 0.

For binary relation R, let $R_\vee$ be disjunctive relation $R_\vee((x_1, x_2), (w_1, w_2)) := R(x_1, w_1) \vee R(x_2, w_2)$. Let $\mathsf{DecompOR}$ be an algorithm that, given a monotone access structure Γ over $[n]$ as input, outputs a monotone access structure Λ over $[2n]$ that $\Gamma_{\boldsymbol{R}_\vee} = \Lambda_{\boldsymbol{R}}$ holds for $\boldsymbol{R}_\vee := (R_\vee^{(1)}, \ldots, R_\vee^{(n)})$ and $\boldsymbol{R} := (R^{(1)}, \ldots, R^{(2n)})$. Let $\Sigma = (\mathcal{C}, \mathcal{Z}, \mathcal{V})$ be a Σ-protocol for R. Let $(\mathsf{Prove}, \mathsf{Verify})$ be a scheme in Figure 1 using Σ. We present our construction of ring signature scheme for monotone access structure in Figure 3.

Theorem 2. *The scheme in Figure 3 is a ring signature scheme for monotone access structure. It is signer anonymous if Σ is witness indistinguishable. It is unforgeable against chosen message and ring attacks if L_R is a hard language, Σ is witness indistinguishable and statistically sound, and hash functions H_c and H_e are non-programmable random oracles for output space 2^μ for sufficiently large μ.*

Proof. Correctness and signer anonymity is almost directly from the completeness and witness indistinguishability of the underlying Σ respectively. Thus we focus on proving unforgeability. Outline of our proof follows that of [**?**].

Game 1: This is the same as the experiment for the chosen message and chosen ring attack. Let G_i be the event that the experiment in Game i outputs 1. We have $\Pr[G_1] = \Pr[\mathsf{Expr}^{\mathsf{euf}}_{\mathsf{RS},\mathcal{A}}(\lambda) = 1]$ by definition.

Let $C \subseteq [n]$ be the index of the corrupted verification keys in the game. Let $\hat{\boldsymbol{vk}}$, $\hat{msg}$, $\hat{\pi}$, and $\hat{\Gamma}$ be the final output from the adversary. Without loss of generality, we assume that $\hat{\boldsymbol{vk}} = \boldsymbol{vk}$ and $\hat{\Gamma}$ is over $[n]$. (The adversary can choose $\hat{\Gamma}$ over a

$\mathsf{RS.KeyGen}(1^\lambda)$:

1. Sample $(x_1, w_1) \leftarrow R_\lambda$ and $(x_2, w_2) \leftarrow R_\lambda$ independently.
2. Output $vk := (x_1, x_2)$ and $sk := (w_1, w_2)$.

$\mathsf{RS.Sign}(\boldsymbol{vk}, \boldsymbol{sk}, msg, \Gamma)$:

1. Parse $\boldsymbol{vk} = (vk^{(1)}, \ldots, vk^{(n)})$ as $\boldsymbol{x} = (x_1, \ldots, x_{2n})$, and $\boldsymbol{sk} = (sk^{(1)}, \ldots, sk^{(n)})$ as $\boldsymbol{w} = (w_1, \ldots, w_{2n})$. (Some $sk^{(i)}$ can be $\perp$. Then $w_{2j} = w_{2j+1} = \perp$.)
2. Run $\Lambda \leftarrow \mathsf{DecompOR}(\Gamma)$.
3. Run $\pi \leftarrow \mathsf{Prove}_\Lambda(\boldsymbol{x}, \boldsymbol{w})$ including msg in all hashings as input.
4. Output π as a signature.

$\mathsf{RS.Verify}(\boldsymbol{vk}, msg, \pi, \Gamma)$:

1. Parse $\boldsymbol{vk} = (vk^{(1)}, \ldots, vk^{(n)})$ as $\boldsymbol{x} = (x_1, \ldots, x_{2n})$.
2. Run $\Lambda \leftarrow \mathsf{DecompOR}(\Gamma)$.
3. Run $b \leftarrow \mathsf{Verify}_\Lambda(\boldsymbol{x}, \pi)$ including msg in all hashings as input.
4. Output b

Fig. 3: Proposed ring signature scheme for access structure Γ.

subset of $[n]$. We can turn such an adversary to one that outputs $\hat{\Gamma}$ as we want.) Let $\hat{\pi}$ be parsed to $\hat{\pi} = \{(\hat{a}_i, \hat{z}_i), \hat{s}_i\}_{i \in [2n]}$. As a valid forgery, it satisfies $C \notin \hat{\Gamma}$. Furthermore, every $(\hat{a}_i, \hat{z}_i), \hat{s}_i$ verifies as $1 = \mathsf{CheckShares}_{\hat{\Lambda}^*}(\hat{s}, \hat{s}_1, \ldots, \hat{s}_{2n})$ for $\hat{s} := H_c(\hat{\Lambda}, \boldsymbol{x}, msg, \hat{a}_1, \ldots, \hat{a}_{2n})$, and $1 = \mathcal{V}_i(\hat{x}_i, \hat{a}_i, \hat{e}_i, \hat{z}_i)$ for $\hat{e}_i := H_e(\hat{\Lambda}, \boldsymbol{x}, msg, i, \hat{s}_i)$ for $i \in [2n]$.

Game 2: We clean up the game by halting at win-by-chance events. As we argued in the proof of soundness, the adversary must make relevant hash queries to the corresponding oracles by itself. As also shown in the same place, there must exist a qualified set A^* in $\hat{\Lambda}$ that, for all $i \in A^*$, $\hat{e}_i := H_e(\hat{\Lambda}, \boldsymbol{x}, msg, i, \hat{s}_i)$ appears after $\hat{s} := H_c(\hat{\Lambda}, \boldsymbol{x}, msg, \hat{a}_1, \ldots, \hat{a}_{2n})$ in the view of the adversary. If any of these are not the case at the end, we let the experiment output 0.

Since these events happen only by chance over the choices of H_c and H_e for large enough domain $\{0,1\}^\mu$, we have $|\Pr[G_2] - \Pr[G_1]| < O(q/2^\mu)$ for at most q times of queries to the random oracles throughout the game.

Game 3: Uniformly choose $i^* \leftarrow [2n]$ and select x_{i^*} as a no-instance, i.e., $x_{i^*} \leftarrow \tilde{L}(\lambda)$ where $\tilde{L}$ is a language that is indistinguishable from L_R and has no intersection with it.

Let $i^{*\mathsf{c}}$ denote $\lfloor i^*/2 \rfloor$, which is the index of the verification key containing x_{i^*}. For now, suppose that $i^{*\mathsf{c}} \notin C$ happens. Answering to the signing queries from the adversary can be done by using the remaining witnesses since they are in a qualified set of Λ. It is perfect due to the WI property of the underlying proofs. If the output distribution of the experiment changes noticeably from that in the previous game, we can construct a successful distinguisher for L_R and $\tilde{L}$.

Let ϵ_{hd} denote the bound for indistinguishability of L_R. We have $|\Pr[G_3 \,|\, i^{*\mathsf{c}} \notin C] - \Pr[G_2]| \leq \epsilon_{\mathsf{hd}}$.

We now evaluate $\Pr[G_3 \,|\, i^{*\mathsf{c}} \notin C]$. Since $C \notin \hat{\Gamma}$, there exists $i^{\dagger} \in A^*$ that $i^{\dagger\mathsf{c}} \notin C$. We have $i^{\dagger} = i^*$ with probability $1/2n$ for uniform i^*. (Note that, for this case, $i^{*\mathsf{c}} \notin C$ holds as well.) For $x_{i^*} \notin L_R$ and fixed $\hat{a}_{i^*}$, probability that challenge $\hat{e}_{i^*}$ uniformly chosen by $H_e(\hat{\Lambda}, \boldsymbol{x}, msg, i^*, \hat{s}_{i^*})$ can have $\hat{z}_{i^*}$ that satisfies $1 = \mathcal{V}(x_{i^*}, \hat{a}_{i^*}, \hat{e}_{i^*}, \hat{z}_{i^*})$ is bound by the statistical soundness error, denoted by ϵ_{st}, of Σ. We thus have $\Pr[G_3 \,|\, i^{*\mathsf{c}} \notin C \wedge i^{\dagger} = i^*] = \frac{1}{2n} \cdot \Pr[G_3 \,|\, i^{*\mathsf{c}} \notin C] < \epsilon_{\mathsf{st}}$.

By accumulating the all above bounds, we have $\Pr[\mathsf{Expr}^{\mathsf{euf}}_{\mathsf{RS},\mathcal{A}}(\lambda) = 1] < O(q/2^{\mu}) + \epsilon_{\mathsf{hd}} + 2n\epsilon_{\mathsf{st}}$ which is negligible if q, n, and μ are polynomials in λ, and ϵ_{hd} and ϵ_{st} are negligible in λ as stated. □

5 Concluding remarks

In this work, we have revisited the CDS composition technique and proposed a modification, that we coin the *share-then-hash* methodology. Our simple technique enhances the previous composition in several flavors, including *more compact* proofs (one single transcript per atomic statement), *better generality* (it is not limited to 2-special sound atomic protocols) and security proofs under *weaker assumptions* (soundness can be proven in the non-programmable random oracle model). Consequently, our results can lead to more efficient, general and secure cryptographic primitives that rely on proofs of partial knowledge.

Proving lower bounds on the proof size and communication complexity of partial proofs of knowledge is an appealing target for future work. In particular, it would be interesting to know if our construction is optimal under some measure or criteria. Another interesting direction for future work would be explore the application of our share-then-hash technique to other scenarios.