Verifiable Secret Sharing from Symmetric Key Cryptography with Improved Optimistic Complexity

Ignacio Cascudo[1], Daniele Cozzo[1], Emanuele Giunta[1,2]

[1] IMDEA Software Institute, Spain.
name.surname@imdea.org
[2] Universidad Politecnica de Madrid, Spain.

Abstract. In this paper we propose verifiable secret sharing (VSS) schemes secure for any honest majority in the synchronous model, and that only use *symmetric-key* cryptographic tools, therefore having plausibly post-quantum security. Compared to the state-of-the-art scheme with these features (Atapoor et al., Asiacrypt '23), our main improvement lies on the complexity of the *"optimistic"* scenario where the dealer and all but a small number of receivers behave honestly in the sharing phase: in this case, the running time and download complexity (amount of information read) of each honest verifier is *polylogarithmic* and the total amount of broadcast information by the dealer is *logarithmic*; all these complexities were linear in the aforementioned work by Atapoor et al. At the same time, we preserve these complexities with respect to the previous work for the "pessimistic" case where the dealer or $O(n)$ receivers cheat actively. The new VSS protocol is of interest in multi-party computations where each party runs one VSS as a dealer, such as distributed key generation protocols.
Our main technical handle is a distributed zero-knowledge proof of low degreeness of a polynomial, in the model of Boneh et al. (Crypto '19) where the statement (in this case the evaluations of the witness polynomial) is distributed among several verifiers, each knowing one evaluation. Using folding techniques similar to FRI (Ben-Sasson et al., ICALP '18) we construct such a proof where each verifier receives polylogarithmic information and runs in polylogarithmic time.

1 Introduction

A (t, n)-threshold secret sharing scheme allows a dealer D to share a secret s among n parties $P_1, \ldots, P_n$ in such a way that any subset of t parties or less has no information about s while any set of $t + 1$ or more parties can recover s if they collaborate. The most famous example of threshold secret sharing is the scheme proposed by Shamir [Sha79]. The dealer samples a polynomial $f(X)$ of degree at most t with coefficients in a finite field $\mathbb{F}$ such that its evaluation in a distinguished point $\alpha_0 \in \mathbb{F}$ is the secret s, i.e. $f(\alpha_0) = s$, and then sends an evaluation $s_i = f(\alpha_i)$ to each party individually, where α_i are pairwise distinct points in $\mathbb{F}$ (and different from α_0). From the basic properties of polynomials it follows that t or less parties do not have any information about s, while $t + 1$

parties can collaboratively reconstruct $f(X)$ and hence recover s by Lagrange interpolation. The above scheme however only provides security against passive adversaries and does not prevent a dishonest dealer from sampling a polynomial of the wrong degree or dishonest parties from sending incorrect values at the moment of reconstruction.

A verifiable secret sharing (VSS) scheme solves these problems by ensuring that the shares parties receive are part of a correct sharing of a secret according to the specified underlying secret sharing scheme, and later that the honest parties reconstruct the secret correctly. VSS is typically used for realizing distributed schemes, for example realizing distributed key generation (DKG), which in turn is a fundamental building block for threshold cryptography [GJKR07], and general purpose multiparty computation (MPC) protocols [BGW88].

VSS schemes have been realized from different assumptions, in both the synchronous and asynchronous model. In this work we focus on the former. On one side, starting with [BGW88] there are VSS that offer perfect security, see [CCP22] for an exhaustive overview. These schemes do not rely on computational hardness assumptions and are typically computationally efficient, but require high communication and assume more than two thirds of parties being honest for reconstructing the secret. On the other hand, computationally secure VSS schemes based on public-key cryptography have been proposed [Fel87, Ped92, Sch99, CD17, CD20, GHL22, CDGK22, Bag23, CD23]. Thanks to public-key tools some of these schemes are able to achieve the property of public verifiability. A publicly verifiable secret sharing (PVSS) is a VSS scheme where all communication is done through a public channel and a public verifier can ensure the correctness of the sharing and reconstruction. This typically employs a linearly homomorphic encryption scheme by which the dealer encrypts the shares and then publishes the ciphertexts, so that parties can use their own secret keys to get the shares. The correctness of the shares is ensured by a proof that the plaintexts of those ciphertexts are indeed shares of a polynomial of degree t, which can be checked by everyone, not only the shareholders. This is particularly useful in some scenarios such as the construction of randomness beacons [CD17, CD20] and multiparty computation in some restricted settings, e.g. in the so-called YOSO model [CDGK22, CD23].

Many of these schemes, in particular all cited above, only require simple honest majority for reconstructing the secret. The downside of this approach is that it suffers from large bandwidth and high cost from the prover due to public-key operations.

A third class of VSS lies in the middle in terms of required security assumptions and attain computational security but only use symmetric-key cryptography. Giving up on perfect secrecy puts these schemes in the regime of honest majority and at the same time relying on symmetric-key tools gives an advantage in terms of efficiency over those relying on public-key primitives. While these schemes are not publicly verifiable, this is enough for many applications, for example multiparty computation and threshold cryptography.

Gennaro, Rabin and Rabin [GRR98] proposed the first computational VSS that relies solely on symmetric-key tools. However their scheme only achieves a weaker notion of security and communication is linear in the number of parties. Backes, Kate and Patra [BKP11] proposed a VSS whose security relies solely on the binding property of commitment schemes, that can be instantiated with hash functions. However, sharing a secret involves using a bivariate polynomial that yields $O(n^2)$ in communication.

The state-of-the-art is a recent work by Atapoor et al. [ABCP23], which uses a novel approach for constructing VSS starting from distributed zero-knowledge (dZK) proofs. The notion of distributed dZK proof was formally introduced in [BBC+19]. In this setting a prover wants to convince n verifiers $V_1, \ldots, V_n$ that a statement x lies in some language $\mathcal{L}$ (in the case of NP-languages that there exists a witness for which $x \in \mathcal{L}$). The main difference with the standard notion of interactive (zero-knowledge) proof is that now x is distributed among the verifiers, but no single verifier knows x in full. A distributed zero-knowledge proof system is required to satisfy correctness, meaning that if all verifiers accept then $x \in \mathcal{L}$; soundness, meaning that if there is no w for which $x \in \mathcal{L}$ then the proof rejects even if the prover colludes with $t-1$ verifiers; and zero-knowledge, namely that the proof does not reveal any information about w to up to t colluding verifiers.

As shown in [ABCP23], one can realize a VSS from a distributed dZK proof. The dealer runs a dZK proof for membership to the following language

$$\mathcal{L} = \{x = (x_1, \ldots, x_n) \in R^n \ : \ \exists f(X) \in R[X] \ : \ x_i = f(\alpha_i), \ \deg(f) \leq t\}. \quad (1)$$

where R is a (commutative, with 1) ring, and $(\alpha_1, \ldots, \alpha_n)$ is a (fixed, public) exceptional set in R.[3] The share-receivers play the role of verifiers for this dZK proof. Then a consensus protocol allows the parties to resolve conflicts in case of rejections, and eventually disqualify the dealer.

The starting point of [ABCP23] is a simple distributed Σ-protocol for (1). At a high level their construction works as follows: the dealer with input a secret polynomial $f(X)$ of degree $\leq t$ with $x_0 = f(\alpha_0)$ and $x_i = f(\alpha_i)$, samples a uniformly random polynomial $b(X)$ also of degree $\leq t$, broadcasts commitments to its evaluations, and then computes a random linear combination $r(X) = b(X) + \mu f(X)$ where $\mu \in \mathbb{F}$ is sampled by the verifiers. The prover then broadcasts the polynomial $r(X)$ while it sends, privately to each party i, the share x_i and the opening to the commitments to $b(\alpha_i)$. Each verifier V_i can individually check the proof by checking the openings of the commitments and that $r(\alpha_i) = b(\alpha_i) + \mu x_i$.

The resulting VSS has a $O(n \log n)$ cost for the dealer, which is inherent due to the need of evaluating a polynomial of degree $O(n)$ on n points. Their VSS works for more general structures than fields, for example rings with a large enough exceptional set. However their protocol requires at least $O(n)$ computational cost for the shareholders already in the sharing phase, as verification requires to evaluate the polynomial $r(X)$. In terms of communication, the dealer

[3] An exceptional set is a set where the pairwise differences of distinct elements in the set are all invertible in the ring

needs to broadcast the whole polynomial $r(X)$ (which amounts to a $O(n)$ amount of broadcast communication) and consequently each verifier has linear download complexity, i.e. each verifier receives $O(n)$ amount of communication.

An important observation for this work is that the above complexities hold *even in the "optimistic" case* where no party eventually acts dishonestly. In particular, the verifiers complexity is $O(n)$ in that case. If there are $\Theta(n)$ corrupt verifiers or if the dealer is corrupt, then the complexity becomes $O(n \log n)$.

This leads us to the question of whether we can design VSS protocols that have a *better optimistic complexity*, i.e., that allow for sublinear $o(n)$ or even polylogarithmic $O(\text{polylog } n)$ verifier work, in the case where the dealer is honest and up to a "small" number (say $O(1)$) of verifiers are corrupted, while still not worsening the verifier complexity of the pessimistic cases where either the dealer and/or a large number of verifiers up to certain bound $t = O(n)$ are corrupted.

Note that even though there is one party (namely, the dealer) who may on their own force the pessimistic case to happen, the scenario above can be of interest in many uses of VSS in multiparty computation protocols where, at a certain round, n instances of VSS are run, one for each of the parties acts as a dealer. One of the most well known examples of this is the case of distributed key generation protocols for discrete-log based threshold schemes, where each party chooses and VSSs a random field element, and the secret key is computed as the sum of the correctly VSSed elements. In cases as the above, if the adversary actively corrupts, say, $O(1)$ parties during the execution, these parties will be able to force the worst case complexity in the $O(1)$ instances where they are acting as dealers, but the remaining instances will enjoy the optimistic complexity. The total work of each honest party across the n VSS instances will be $O(n \text{ polylog } n)$, which would be an improvement over the $O(n^2)$ complexity that would arise from using [ABCP23]. At the same time, the VSS would still remain secure against a more powerful adversary corrupting $O(n)$ parties.

While verifier complexity is our main concern in this paper, we are also interested in improving the *download complexity* per party and *broadcast communication* in these optimistic cases. The latter is interesting in blockchain ecosystems, where one wants to limit as much as possible the amount of information stored on-chain. We do need to remark that this will come at the cost of increasing the amount of private communication sent by the dealer to each party, as we describe in the next paragraphs.

Our contributions. In this work we present a VSS construction that only employs secret key cryptography, tolerates $t < n/2$ corrupt parties, and improves on the state of the art with regards to *optimistic verifier complexity* as well as optimistic broadcast and download complexity, while still matching the complexities of previous works in the pessimistic case. In particular, in the optimistic case where the dealer is honest and there are $O(1)$ corrupt verifiers, the verifier complexity is $O(\log(n)^2)$, the dealer broadcasts $O(\log n)$ information and each party needs to receive $O(\log(n)^2)$ information in total, via the broadcast and private channels. All these complexities were $O(n)$ in [ABCP23]. In the pessimistic case (where the dealer and/or $O(n)$ verifiers are corrupt) the asymptotic complexities match

those of [ABCP23]: the broadcast and download complexities are $O(n)$ and the verifier complexity is $O(n \log n)$. In all cases the prover complexity is $O(n \log n)$, same as in [ABCP23]. This all comes at the cost of increasing the private communication: while in [ABCP23] the dealer needs to send a constant amount of information privately to each party, in our work this will be $O(\log(n)^2)$. While this increases the total amount of information communicated by the dealer, the decreased use of the broadcast channel may be beneficial in some applications as we have argued above.

Our main technical handle is a new distributed ZK proof for language (1) with polylogarithmic proof size in the degree of the polynomial that only leverages on symmetric-key primitives and supports rings with a large enough exceptional set, and which we believe to be of independent interest.

In more detail, in our VSS protocol the dealer initially broadcasts $O(\log n)$ information (the "public" part of the aforementioned distributed ZK proof), as well as sending $O(\log(n)^2)$ information to each party privately, which contains the shares and private part of the distributed proof. The verifiers are required to perform $O(\log(n)^2)$ computation[4]. Only if there are complaints, the protocol incurs in more communication and computation as the prover then needs to broadcast the private communication previously sent to the complaining parties. In the worst case where $O(n)$ share receivers complain, the dealer needs to broadcast $O(n)$ information (and no additional private communication) and the share receivers need to perform $O(n \log n)$ computation. As mentioned above, these costs are the same as in the state-of-the-art [ABCP23], with the one aforementioned caveat that we require more private communication. See Table 1 for comparisons.

Our VSS tolerates up to $t = n/2 - 1$ corruptions and is proven secure in the random oracle model and, as in [ABCP23], we are able to support rings with a large enough exceptional set.

Technical overview of our approach. To construct our VSS, we follow the paradigm of [ABCP23] starting from a distributed ZKP of the existence of a low degree polynomial interpolating the shares. At the heart of our work is an efficient distributed proof (without zero-knowledge) for the same task.

More precisely, the n verifiers each have a piece $x_i \in R$ of the statement, and the prover wishes to convince them that there is a polynomial $f(X)$ of degree $< d$ such that $x_i = f(\alpha_i)$ for all $i \in \{1, \ldots, n\}$ (or more precisely, for all the honest verifiers V_i).

Our interactive proof is recursive and based on a folding technique, similar to proofs in the literature such as FRI [BBC+19] and DARK compilers [BFS20]. At every step k of the recursion the prover claims that a certain polynomial $f^{(k)}(X)$ has degree $< d/2^k$ (where in addition $f^{(0)} = f$); the recursion reduces this task to proving that some related *randomized* polynomial $f^{(k+1)}(X)$ has degree $< d/2^{k+1}$. At the last step $k = \tau$ of the recursion, the prover simply broadcasts $f^{(\tau)}$ which is of small enough degree $d/2^\tau$. For the folding of $f^{(k)}$

[4] Specifically $O(\log n)$ ring operations and $O(\log(n)^2)$ hashes

into $f^{(k+1)}$ we offer two alternatives: in our first alternative, which is inspired by [BFS20], the prover splits $f^{(k)}$ in high and low degree terms (i.e. $f^{(k)}(X) = g_0^{(k+1)}(X) + X^{d/2^{k+1}} g_1^{(k+1)}(X)$), receives a random challenge $\mu^{(k)}$ and constructs $f^{(k+1)}$ as $f^{(k+1)}(X) = g_0^{(k+1)}(X) + \mu_1 g_1^{(k+1)}(X)$. The second alternative is the one in FRI: the prover splits $f^{(k)}$ in odd and even degree terms, i.e. as $f^{(k)}(X) = g_0^{(k)}(X^2) + X g_1^{(k)}(X^2)$, and again sets $f^{(k+1)}(X) = g_0^{(k+1)}(X) + \mu_1 g_1^{(k+1)}(X)$.

The reason why we have two alternatives is that, for general rings R, the first alternative requires fewer assumptions on the set of evaluation points $\{\alpha_1, \ldots, \alpha_n\}$; namely, we require that this is an exceptional set, i.e. that the pairwise differences of all the elements in the set are invertible in the ring. On the other hand, using the second alternative requires in addition that *all* sets $\{\alpha_1^{2^k}, \ldots, \alpha_n^{2^k}\}$ for $k = 0, \ldots, \tau$ are exceptional. However, if we do have this guarantee, for example if R is a finite field, the second alternative may also lead to more efficient protocols.

A technical difference with FRI is in how the prover shows that this splitting has been done correctly. Instead of committing to the evaluation of $f^{(k)}(X)$, $g_0^{(k+1)}$, $g_1^{(k+1)}$ in a large set of points and then opening a random subset chosen by the (single) verifier, in our case the prover commits to the evaluations of these polynomials in $\{\alpha_1, \ldots, \alpha_n\}$ (or $\{\alpha_1^{2^k}, \ldots, \alpha_n^{2^k}\}$ in the second alternative) and then opens the evaluations in the i-th point privately to the i-th verifier. Note that in our case, the prover cannot cheat, even with small probability, in this step as essentially the honest verifiers are checking that the splitting is correct in all honest points. The only source of soundness error in our case is the fact that the degree of $g_0^{(k+1)}(X) + \mu_1 g_1^{(k+1)}(X)$ may be smaller than the degrees of both $g_0^{(k+1)}(X)$ and $g_1^{(k+1)}(X)$, which happens with small probability if μ is sampled from a large exceptional set of the ring.

The recursive nature of our protocol allows us to achieve a proof where the size of the communication received by each verifier (both broadcast and privately) is polylogarithmic in the degree of the polynomial. Instantiating the commitments with Merkle trees allows for efficient openings and give the protocol computational security based only on the security of hash functions.

Our technique allows us to work over rings with a large enough exceptional set, and we believe this to be of independent interest as it would allow constructing ring-friendly polynomial commitments that are plausibly post-quantum secure.

We then show how to easily add zero-knowledge to the protocol above by adding two additional rounds. In the first round the prover samples a uniformly random polynomial $b(X)$ of the same degree of $f(X)$ and sends a commitment to it to the verifiers, that will respond with a random challenge μ_0. The prover then computes the random linear combination $r(X) = b(X) + \mu_0 f(X)$ and applies to it the above folding protocol. The zero knowledge property comes from the fact that $b(X)$ information-theoretically hides $f(X)$.

By interpreting these two additional rounds as the first two rounds of the distributed Σ-protocol of [ABCP23], we can see our construction as a dis-

tributed version of the technique used to compress standard Σ-protocols [AC20] where the prover replaces the third message of a Σ-protocol by a (non zero-knowledge) proof that this last message satisfies a certain property (in this case that $\deg r(X) < d$). We then turn the above $(2\tau + 3)$-rounds dZK proof into a non-interactive dZK proof using the Fiat-Shamir transform.

Following the construction of [ABCP23], we obtain a VSS by adding a consensus protocol on the execution of the non-interactive dZK proof. The resulting VSS inherits the logarithmic communication complexity and computational costs from the dZKP.

Comparison with previous work. We compare our VSS with other honest majority schemes in Table 1. For [ABCP23] and our work we include the costs for both the optimistic (no complaints) and pessimistic ($O(n)$ complaints) case. Such a distinction does not exist in the case of PVSS thanks to the public verifiability feature. Such a feature, however, comes at the price of having $O(n)$ broadcast communication and $O(n \log n)$ computational complexity for both dealer and parties in terms of (expensive) group operations. Compared to the state-of-the-art [ABCP23], in the optimistic case we decrease the amount of information to be broadcasted from $O(n)$ to $O(\log n)$, as well as the total download per party, from $O(n)$ to $O(\log(n)^2)$, and parties computational cost from $O(n)$ to $O(\log n)$. Instead we add a polylogarithmic factor $\log(n)^2$ to the amount of private communication. The computational cost for the dealer is the same, $O(n \log n)$ ring operations, which is an inherent cost from the evaluation of the secret polynomial defining the shares. In the pessimistic case, the cost are the same, except we still pay for the polylogarithmic factor in private communication.

Other related work. In this work we only focus on protocols that operate in the so-called synchronous model. Here the parties are synchronized by a global clock and there are strict (publicly-known) upper bounds on the message delays. On the other hand, there is a line of works that explores protocols in the asynchronous model [AKP20, CP23, AJM+23, SS23], where the parties are not synchronized and where the we assume that the adversary can take control of the network and arbitrarily delay the messages sent by the parties. Designing VSS protocols in this setting is more challenging and inherently support at most $n/3 - 1$ corruptions. Our techniques, in particular, do not apply to this asynchronous case, where parties would need to broadcast and read decision bits (even in the case everyone accepts) so communication and verification time would be linear even in the optimistic case.

Outline. In Section 2 we recall and revise known definitions. Our distributed proof of low degree is detailed in Section 3, starting from two (non ZK) protocols (Sections 3.1-3.2) and later adding zero knowledge (Section 3.3) and removing interaction (Section 3.4). Finally, Section 4 is devoted to building VSS from distributed proofs of low degree.

Table 1. Comparison of our VSS with previous computationally secure VSS For comparison, we specialize our scheme and that of [ABCP23] to $R = \mathbb{F}$, a finite field. PV: public verifiability, BC: broadcast, PC: private communication to each party, DW: download per party, $O^{\mathbb{F}}(\bullet)$: complexity in terms of field operations, $O^{\mathbb{G}}(\bullet)$: complexity in terms of group operations. Note that publicly verifiable secret sharing schemes assume an initial PKI setup, or otherwise need an additional round to establish this PKI.

Scheme	Assumption	Rounds	Prover complexity	Verifier complexity	Communication	PV
[Sch99]	DDH, RO	1	$O^{\mathbb{G}}(n \log n)$	$O^{\mathbb{G}}(n^2 \log n)$	BC: $O(n)$ PC: $-$ DW: $O(n)$	yes
[CD17, CD20], [CDGK22]	DDH, RO	1	$O^{\mathbb{G}}(n \log n)$	$O^{\mathbb{G}}(n \log n)$	BC: $O(n)$ PC: $-$ DW: $O(n)$	yes
[CD23, KMM$^+$23]	Class group assump., RO	1	$O^{\mathbb{G}}(n \log n)$	$O^{\mathbb{G}}(n \log n)$	BC: $O(n)$ PC: $-$ DW: $O(n)$	yes
[ABCP23] optimistic case	SK, RO	2	$O^{\mathbb{F}}(n \log n)$	$O^{\mathbb{F}}(n)$	BC: $O(n)$ PC: $O(1)$ DW: $O(n)$	no
[ABCP23] pessimistic case	SK, RO	3	$O^{\mathbb{F}}(n \log n)$	$O^{\mathbb{F}}(n \log n)$	BC: $O(n)$ PC: $O(1)$ DW: $O(n)$	no
This work optimistic case	SK, RO	2	$O^{\mathbb{F}}(n \log n)$	$O^{\mathbb{F}}(\log^2 n)$	BC: $O(\log n)$ PC: $O(\log^2 n)$ DW: $O(\log^2 n)$	no
This work pessimistic case	SK, RO	3	$O^{\mathbb{F}}(n \log n)$	$O^{\mathbb{F}}(n \log n)$	BC: $O(n)$ PC: $O(\log^2 n)$ DW: $O(n)$	no

2 Preliminaries

2.1 Notation

We denote by $[n]$ the set $\{1, \ldots, n\}$. In what follows R denotes a ring and $R[X]$ (resp. $R[X]_t$) the ring of polynomials (resp. degree $\leq t$ polynomials) with coefficients in R. Vectors are denoted in bold. For a vector $\mathbf{x}$ we denote by x_i its i-th entry, while $\mathbf{x}_H$ denotes the subvector $(x_i)_{i \in H}$ for a given set of indices H. All logarithms are assumed in base two.

2.2 Adversarial and communication model

In our protocols we will consider a network of n parties $P_1, \ldots, P_n$. We work in the synchronous communication model and assume that parties are conncted to each other by private, authenticated and bidirectional channels.

We further assume that all parties have access to a broadcast channel [PSL80]. This means that parties can send a message reliably to each other. In addition,

if a party receives a message via a broadcast, then it knows that all other honest parties received the same value.

A synchronous network allows protocols to operate in a sequence of rounds. In each round, parties perform some local computation, send messages (if any) through the private and authenticated link, and broadcast some information over the broadcast channel. At the end of each round, they receive all messages sent or broadcast by the other parties in the same round.

For the adversarial model, we assume a static, malicious adversary that corrupts up to $t < \frac{n}{2}$ parties in the protocol. While honest parties send messages following the protocol, corrupt parties may send arbitrary messages. Also, the adversary may be rushing, meaning that at each round of the protocol it waits to see what the other parties have broadcasted before broadcasting its own messages.

2.3 Vector commitments

Definition 1. *A vector commitment with message space* $\mathsf{VC}.M$ *and commitment space* $\mathsf{VC}.C$ *is a tuple of algorithms* $\mathsf{VC} = (\mathsf{VC.Setup}, \mathsf{VC.Com}, \mathsf{VC.Open}, \mathsf{VC.Vfy})$ *defined as follows:*

- $\underline{\mathsf{pp} \leftarrow \mathsf{VC.Setup}(1^\lambda, n)}$ *: given the security parameter and size of the vectors to be committed returns public parameters.*
- $\underline{(\mathsf{cm}, \mathsf{aux}) \leftarrow \mathsf{VC.Com}(\mathsf{pp}, \mathbf{x})}$ *: given public parameters and a vector* $\mathbf{x} \in \mathsf{VC}.M^n$ *outputs a commitment* cm *and auxiliary information* aux *used for opening.*
- $\underline{\mathsf{op} \leftarrow \mathsf{VC.Open}(\mathsf{pp}, \mathsf{cm}, i, \mathsf{aux})}$ *: given public parameters, an index* $i \in [n]$ *and a commitment* cm *with auxiliary information* aux*, returns an opening proof* op *for position* i*.*
- $\underline{b \leftarrow \mathsf{VC.Vfy}(\mathsf{pp}, \mathsf{cm}, x, i, \mathsf{op})}$ *: given public parameters, an element* x*, position* i*, proof* op *and a commitment* cm*, it checks the validity of the opening proof.*

Properties that a secure VC is required to satisfy are correctness, position binding and succinctness. In this work we further consider *hiding* VC.

Correctness. A vector commitment is correct if for any $\mathbf{x} \in \mathsf{VC}.M^n$

$$\Pr\left[\mathsf{VC.Vfy}(\mathsf{pp}, \mathsf{cm}, x_i, i, \mathsf{op}) = 1 \;:\; \begin{array}{l} \mathsf{pp} \leftarrow \mathsf{VC.Setup}(1^\lambda) \\ \mathsf{cm}, \mathsf{aux} \leftarrow \mathsf{VC.Com}(\mathsf{pp}, \mathbf{x}) \\ \mathsf{op} \leftarrow \mathsf{VC.Open}(\mathsf{pp}, \mathsf{cm}, i, \mathsf{aux}) \end{array}\right] = 1.$$

Position binding. Define the advantage:

$$\begin{aligned} &\mathrm{Adv}^{\mathrm{PBinding}}_{\mathsf{VC}}(\mathcal{A}) = \\ &= \Pr\left[\begin{array}{l} \mathsf{VC.Vfy}(\mathsf{pp}, \mathsf{cm}, x_0, i, \mathsf{op}_0) = 1 \\ \mathsf{VC.Vfy}(\mathsf{pp}, \mathsf{cm}, x_1, i, \mathsf{op}_1) = 1 \end{array} \;:\; (\mathsf{cm}, i, x_0, \mathsf{op}_0, x_1, \mathsf{op}_1) \leftarrow \mathcal{A}(\mathsf{pp}) \right]. \end{aligned}$$

Then a vector commitment VC satisfies position binding if, for all honestly generated public parameters pp, for all PPT adversaries $\mathcal{A}$ one has

$$\mathrm{Adv}^{\mathrm{PBinding}}_{\mathsf{VC}}(\mathcal{A}) = \mathsf{negl}(\lambda).$$

Hiding. The hiding property [Giu23] informally states that opening a number of positions in a vector commitment should keep the unopened ones hidden. Formally, for any non-empty subset $T \subset [n]$ define the advantage

$$\mathrm{Adv}^{\mathrm{Hiding}}_{\mathsf{VC}}(\mathcal{A}, T) \;=\; \Pr\left[b = b' \;:\; \begin{array}{l} \mathsf{pp} \leftarrow \mathsf{VC.Setup}(1^\lambda),\ b \leftarrow \{0,1\} \\ \mathbf{x}_0, \mathbf{x}_1 \leftarrow \mathcal{A}(\mathsf{pp}),\ \mathbf{x}_0[j] = \mathbf{x}_1[j]\ \forall j \in T \\ (\mathsf{cm}, \mathsf{aux}) \leftarrow \mathsf{VC.Com}(\mathsf{pp}, \mathbf{x}_b) \\ \mathsf{op}_j \leftarrow \mathsf{VC.Open}(\mathsf{pp}, \mathsf{cm}, j, \mathsf{aux})\ \forall j \in T \\ b' \leftarrow \mathcal{A}(\mathsf{pp}, \mathsf{cm}, \{\mathsf{op}_j\}_{j \in T}) \end{array} \right].$$

Then we say that VC is hiding if, for all non-empty $T \subset [n]$ and all PPT adversaries $\mathcal{A}$ one has that

$$\mathrm{Adv}^{\mathrm{Hiding}}_{\mathsf{VC}}(\mathcal{A}, T) = \frac{1}{2} + \mathsf{negl}(\lambda).$$

Succintness. A vector commitment is succint if both the commitment cm and opening proof op for a position i are independent of the size of the committed vector. In this work, we will slightly relax this notion, by allowing the opening proof for a position to be *logarithmic* in the size n of the vector. For example, this is the case of Merkle trees, which are discussed next.

Merkle trees. In this work we will be using Merkle trees to instantiate (hiding) vector commitments. [5] For Merkle trees we will be using the specific notation $\mathsf{MT} = (\mathsf{MT.Setup}, \mathsf{MT.Com}, \mathsf{MT.Open}, \mathsf{MT.Vfy})$. Let n be a power-of-two[6] and let H be a collision resistant hash function. A Merkle tree commitment to a vector $\mathbf{x}$ of n elements consists in a binary tree, where the leaves are the hash of the elements x_i and each node is the hash of its children. It is a tree of height $\log n$. Formally, setting $d = \log n$ if we label the entries of $\mathbf{x}$ with the binary representation of their indexes then

$$\begin{array}{rr} \text{leaves:} & h_{b_0 b_1 \ldots b_d} = H(x_{b_0 b_1 \ldots b_d}) \\ \text{level 1:} & h_{b_0 b_1 \ldots b_{d-1}} = H(h_{b_0 b_1 \ldots b_{d-1} 0} \| h_{b_0 b_1 \ldots b_{d-1} 1}) \\ \text{level } k\text{:} & h_{b_0 b_1 \ldots b_{d-k}} = H(h_{b_0 b_1 \ldots b_{d-k} 0} \| h_{b_0 b_1 \ldots b_{d-k} 1}) \end{array}$$

A public commitment to $\mathbf{x}$ then consists in the root of such a tree, with the convention that the root is indexed by the empty string ϵ, namely $\mathsf{cm} = h_\epsilon =$

[5] The reason being that Merkle trees can be realized from hash functions only which is fundamental to our purpose of realizing a VSS from symmetric-key cryptography only. Furthermore they do not require setup.

[6] If not we can pad the vector with zeros.

$H(h_0 \| h_1)$. Opening the commitment at position i can be done by revealing the corresponding leaf x_i and all the sibling values of all the nodes in the path from x_i till the root, which is logarithmic in the size of $\mathbf{x}$.

Formally, for a bit b, let $\overline{b}$ denote $1 - b$. Then an opening for position i with binary representation $b_0 \dots b_d$ is given by the list

$$\left(x_{b_0 \dots b_{d-1} b_d}, h_{b_0 \dots b_{d-1} \overline{b_d}}, h_{b_0 \dots b_{d-2} \overline{b_{d-1}}}, \dots, h_{b_0 \dots b_{d-k} \overline{b_{d-k+1}}}, \dots, h_{\overline{b_0}}\right).$$

Verifying the opening requires hashing the siblings to recompute the nodes in the path, and checking that the last node is indeed the same as the initial commitment root. The position binding of the Merkle tree can be reduced to the collision resistance of H.

It is possible to turn such a construction into an hiding vector commitment by computing the leaves using a uniformly random string from $\{0,1\}^\lambda$, one for each leaf. In other words $h_{b_0 b_1 \dots b_d} = H(x_{b_0 b_1 \dots b_d} \| r_{b_0 b_1 \dots b_d})$. Then an opening for position i must include r_i. For simplicity, in the rest of the paper we will assume that all Merkle trees are hiding.

2.4 Distributed zero-knowledge proofs

Here we recall the definition of zero-knowledge proof for distributed relations, introduced in [BBC+19], suitably adapted to rings.

Definition 2 (Distributed Inputs, Languages, and Relations [BBC+19]). *Let n be a number of parties, R be a ring, and $l, l_1, l_2, \cdots, l_n \in \mathbf{N}$ be length parameters, where $l = l_1 + l_2 + \cdots + l_n$. An n-distributed input over R^l (or just distributed input) is a vector $x = x^{(1)} \| x^{(2)} \| \cdots \| x^{(n)} \in R^l$ where each $x^{(i)} \in R^{l_i}$ is called a piece (or share) of x. An n-distributed language $\mathcal{L} \subset R^l$ is a set of n-distributed inputs. A distributed NP relation with witness length h is a binary relation $\mathcal{R} \subset R^l \times R^h$ of pairs (x, w) with n-distributed input $x \in R^l$ and witness $w \in R^h$, such that $(x, w) \in \mathcal{R}$ can be checked in polynomial time. Finally, we let $\mathcal{L}_\mathcal{R} = \{x \in R^l : \exists w \in R^h,\ (x, w) \in \mathcal{R}\}$.*

Definition 3 (n-Verifier Interactive Proofs [BBC+19]). *An* n-Verifier Interactive Proof *protocol over R is an interactive protocol $\Pi = (P, V_1, V_2, \cdots, V_n)$ involving a prover P and n verifiers $V_1, V_2, \cdots, V_n$. The protocol proceeds as follows.*

- *In the beginning of the protocol the prover holds an n-distributed input $x = x^{(1)} \| x^{(2)} \| \cdots \| x^{(n)} \in R^l$, a witness $w \in R^h$, and each verifier V_j holds an input piece (or share) $x^{(j)}$.*
- *The protocol allows the parties to communicate in synchronous rounds over secure point-to-point channels and a broadcast channel.*
- *At the end of the protocol each verifier outputs either 1 (accept) or 0 (reject) based on its view, where the view of V_j consists of its input piece $x^{(j)}$, its random input $r^{(j)}$, and messages it received during the protocol execution.*

The protocol accepts if all verifiers accept, and the protocol rejects if at least one verifier rejects.[7]

In the following, $\Pi(x, w)$ denotes running Π on shared input x and witness w, and says that $\Pi(x, w)$ accepts (respectively, rejects) if at the end all verifiers (resp. at least one honest verifier) output 1 (resp., 0). $View_{\Pi,T}(x, w)$ denotes the (joint distribution of) views of verifiers $\{V_j\}_{j\in T}$ in the execution of Π on distributed input x and witness w.

Let $\mathcal{R}(x, w)$ be an n-distributed relation over a ring R. We say that an n-verifier interactive proof protocol $\Pi = (P, V_1, \cdots, V_n)$ is a distributed strong ZK proof protocol for R with t-security against malicious prover *and* malicious verifiers, and with soundness error ε, if Π satisfies the following properties:

Definition 4 (Correctness). *For every n-distributed input $x = x^{(1)}\|x^{(2)}\|\cdots\|x^{(n)} \in R^l$ and $w \in R^h$ such that $(x, w) \in \mathcal{R}$, the execution of $\Pi(x, w)$ accepts with probability 1. Note this definition assumes the prover and all verifiers behave honestly.*

Definition 5 (ε-Soundness Against Prover and t Verifiers). *For every $T \subseteq [n]$ of size $|T| \leq t$, an adversary $\mathcal{A}$ controlling the prover P and verifiers $\{V_j\}_{j\in T}$, n-distributed input $x = x^{(1)}\|x^{(2)}\|\cdots\|x^{(n)} \in R^l$, and every $w \in R^h$, the following holds. If there is no n-distributed input $x' \in \mathcal{L}_{\mathcal{R}}$ such that $x'_H = x_H$, where $H = [n] \setminus T$, the execution of $\Pi^\star$ rejects except with probability at most ε, where $\Pi^\star$ denotes the interaction of $\mathcal{A}$ with the honest verifiers.*

In analogy to ordinary interactive proofs, we say soundness holds *adaptively* if the input is chosen by the malicious prover (see for instance [AFK22]), potentially after observing the public parameters, or interacting with the random oracle.

Definition 6 (Strong Zero-Knowledge against t Verifiers). *For every $T \subseteq [n]$ of size $|T| \leq t$ and a malicious adversary $\mathcal{A}$ controlling the verifiers $\{V_j\}_{j\in T}$, there exists a simulator Sim such that for every n-distributed input $x = x^{(1)}\|x^{(2)}\|\cdots\|x^{(n)} \in R^l$, and witness $w \in R^h$ such that $(x, w) \in \mathcal{R}$, we have $\mathsf{Sim}((x^{(j)})_{j\in T}) \equiv View_{\Pi^\star,T}(x, w)$. Here, $\Pi^\star$ denotes the interaction of adversary $\mathcal{A}$ with the honest prover P and the honest verifiers $\{V_j\}_{j\in[n]\setminus T}$.*

In order to later provide a compiler analogous to the Fiat-Shamir transform, we define a distributed proof to be *public coin* if in the interactive phase verifiers only executes a coin-tossing protocol $\mathcal{F}_{coin}$. Finally we also consider the stronger notion of *round-by-round soundness* [CCH+18], adapted to the distributed proof setting and restricted for simplicity to the public coin case.

[7] We stress the fact that in our definition, the protocol rejects if at least one verifier rejects, unlike [BBC+19], where the protocol rejects if all verifiers reject. Although the latter is a stronger notion of soundness, it is always possible to achieve by adding some rounds of consensus among the verifiers. Later we'll construct a VSS from a ZK proof on distributed inputs. This is done by adding such a consensus protocol outside the proof system. We made this choice so as to mark clearly the step from proof system to VSS.

Definition 7. *A distributed public coin proof has ε-round-by-round soundness against a prover and t verifiers if there exists a set D of doomed transcripts such that, for any $T \subseteq [n]$ of size $|T| \leq t$*

1. *Given $(x_i)_{i \notin T}$ such that $\nexists x' \in \mathcal{L}_{\mathcal{R}} : x'_i = x_i$ for $i \notin T$, then $(x_i, \varnothing)_{i \notin T} \in D$.*
2. *Given $(x_i, v_i)_{i \notin T} \in D$, where v_i denotes the view of verifier V_i till the current state, for any reply (M, m_i) where M, m_i denote the messages broadcast and privately sent to V_i respectively at the end of the current round, and random coins μ tossed by the verifiers*

$$\Pr\left[(x_i, (v_i \| M \| m_i \| \mu))_{i \notin T} \notin D\right] \leq \varepsilon(\lambda).$$

3. *For any list of full transcripts[8], $(x_i, v_i)_{i \notin T} \in D \;\Rightarrow\; \exists j \notin T : V_j$ rejects.*

2.5 Interpolation and Shamir secret sharing over rings

A (t, n)-Shamir secret sharing scheme allows n parties to individually hold a share x_i of a common secret x_0, such that any subset of t parties or less are not able to learn any information about the secret x_0, while any subset of at least $t+1$ parties are able to efficiently reconstruct the common secret x_0. While Shamir's scheme is originally defined over a finite field $\mathbb{F}$, meaning the secret and all shares are values in $\mathbb{F}$, it can be extended to rings through exceptional sets, as described in [CDN15]. We recall the details here.

Definition 8. *Let R be a ring. An* exceptional set *is a set $S \subset R$, where for every pair $x, x' \in S$ with $x \neq x'$, the difference $x - x'$ is invertible in R.*

Lemma 1. *Let $m > t \geq 0$ be integers, R a ring, $S = \{\alpha_1, \ldots, \alpha_m\} \subseteq R$ an exceptional set. Then for every $Q = \{i_1, \ldots, i_{t+1}\} \subseteq [m]$ the map $R[X]_t \to R^{t+1}$ given by $f(X) \mapsto (f(\alpha_{i_1}), \ldots, f(\alpha_{i_{t+1}}))$ is an R-module isomorphism.*

In details, the inverse of the isomorphism above is given by mapping $(x_1, \ldots, x_{t+1})$ to $f(X) = \sum_{i \in Q} x_i \cdot L_i^Q(X)$ where

$$L_i^Q(X) := \prod_{j \in Q \setminus \{i\}} \frac{\alpha_j - X}{\alpha_j - \alpha_i}$$

are the Lagrange basis polynomials, which are all of degree t, and well defined thanks to Q being exceptional.

Now (t, n)-Shamir secret sharing can be defined over R, as long as it contains an exceptional set of "evaluation points" $\mathcal{E} = \{\alpha_0, \alpha_1, \ldots, \alpha_n\} \subset R$ of size $n+1$. Each party P_i, $i \in [n]$, is associated to the element α_i while α_0 is associated to the secret. To share secret x_0, a polynomial $f(x)$ is chosen uniformly at random in the set of polynomials in $R[X]_t$ with $f(\alpha_0) = x_0$. Each party P_i is assigned the secret share $x_i = f(\alpha_i)$. Then any subset $Q \subseteq \{1, \ldots, n\}$ of at least $t+1$ parties can reconstruct the secret x_0 via Lagrange interpolation by computing

[8] The transcript until the point when the verifier halts.

$x_0 = f(\alpha_0) = \sum_{i \in Q} x_i \cdot L_i^Q(\alpha_0)$, where L_i^Q is as above. Moreover, also based on Lemma 1, a subset of t or fewer parties are not able to find $x_0 = f(\alpha_0)$, as this is information theoretically hidden from the other shares. See [CDN15] for details.

Finally we will need another technical lemma involving exceptional sets, which can be derived easily from Lemma 1.

Lemma 2. *Let $S \subset R$ be an exceptional set. Let $N \geq 1$ be an integer and $h^{(0)}(X)$, $h^{(1)}(X), \ldots, h^{(N)}(X)$ be $N+1$ arbitrary polynomials in $R[X]$. Let $d := \max\left\{\deg h^{(i)}(X) : i \in \{0, \ldots, N\}\right\}$.*

Then if $\nu_1, \ldots, \nu_N$ are sampled independently and uniformly at random in S,

$$\Pr\left[\deg\left(h^{(0)}(X) + \sum\nolimits_{i=1}^{N} \nu_i h^{(i)}(X)\right) < d\right] \leq \frac{1}{|S|}$$

Proof. If $d > \deg h^{(i)}(X)$ for all $i \in \{1, \ldots, N\}$ (hence also $\deg h^{(0)} = d$), then the claim is trivial.

Otherwise there exists a (possibly non-unique) $\ell \in \{1, \ldots, N\}$ such that $\deg h^{(\ell)} = d$. We show that for *every* fixed choice of $(\nu_i)_{i \neq \ell} \in S^{N-1}$, there exists at most one $\nu_\ell \in S$ such that $\deg(h_0(X) + \sum_{i=1}^{N} \nu_i h^{(i)}(X)) < d$. This is clearly enough to show the claim.

Let $f(X) = h^{(0)}(X) + \sum_{i=1, i \neq \ell}^{N} \nu_i h^{(i)}(X)$. Note $f(X) + \nu_\ell h^{(\ell)}(X)$ is the polynomial whose degree we want to bound. Let f_d, $h_d^{(\ell)}$ respectively denote the coefficients of X^d in $f(X)$ and $h^{(\ell)}(X)$ (if $\deg f < d$ then $f_d = 0$). Note that $h_d^{(\ell)} \neq 0$. Let $m(X) = f_d + h_d^{(\ell)} X \in R[X]_1$. There is at most one value $s \in S$ such that $m(s) = 0$, otherwise if $m(s') = 0$ for some other $s' \in S$, then $m(X)$ and 0 would be two different polynomials in $R[X]_1$ with the same evaluations in a set $\{s, s'\} \subset S$ of size 2, which is impossible by Lemma 1. Therefore there is at most one value $\nu_\ell = s$ in S for which the coefficient of X^d in $f(X) + \nu_\ell h^{(\ell)}(X)$ is 0, and hence for which this polynomial can have degree less than d.

2.6 Verifiable secret sharing scheme

Definition 9 (From [BKP11, CCP22]). *A (t, n)-VSS protocol is an interactive protocol between n parties $P_1, \ldots, P_n$ and a distinguished party, the dealer, denoted by D and consists of two phases, a sharing phase and a reconstruction phase, defined as follows:*

1. Share*: initially D holds an input x_0, referred to as the secret. The sharing phase may consist of several rounds of interaction between the parties. At the end of the sharing phase, each honest party P_i holds a view v_i that may be used later to reconstruct the dealer's secret.*
2. Reconstruction*: in this phase each party P_i publishes its entire view v_i from the sharing phase, and a reconstruction algorithm* Reconstruction$(v_1, \ldots, v_n)$ *is run and the output is taken as the protocol output.*

Definition 10. *A (t, n)-VSS is secure if for every adversary that controls parties $\{P_i\}_{i \in T}$ belonging to a subset $T \subseteq [n]$ of size $|T| \leq t$, possibly including the dealer, it satisfies the following properties up to negligible probability.*

1. ***Correctness***. *If the dealer is honest (i.e., not controlled by the adversary), then all honest parties output x_0 at the end of* Reconstruction.
2. t-***Privacy***. *If the dealer is honest, then the adversary's view at the end of* Share *reveals no information about the secret x_0. In other words, the adversary's view is identically distributed for all possible secrets x_0.*
3. ***Commitment***. *If the dealer is dishonest (i.e. controlled by the adversary), then at the end of* Share *there exists a unique value $x_0^* \in R \cup \{\perp\}$ such that at the end of* Reconstruction *all parties return x_0^*.*

A stronger notion of VSS requires correctness, t-privacy and strong commitment defined as follows:

3. ***Strong commitment***. *The scheme has the commitment property above and in addition, if the dealer is dishonest, at the end of the sharing phase each (honest) party locally outputs a share of the secret chosen only in R, such that the joint shares output by honest parties are consistent with a specified secret sharing scheme.*

In this work we will focus on Shamir secret sharing schemes. Then the definition of strong commitment means that at the end of the sharing phase, the shares x_i held by the honest parties implicitly define a polynomial $f(X) \in R[X]_t$ of degree t such that $f(\alpha_i) = x_i$.

Our constructions will actually only achieve a computational flavor of privacy. In this case, we argue that for any two different secrets x_0, x_0^*, the distributions of the views of any adversary corrupting at most t share-receivers in respectively a sharing of x_0 and x_0^* are computationally indistinguishable. We capture this by stating that for any such an adversary there is a simulator that can produce a view that is computationally indistinguishable of the sharing of any secret x_0.

Definition 11. ***Computational t-privacy***. *For any adversary $\mathcal{A}$ corrupting a set of parties $\{P_i\}_{i\in T}$, with $|T| \leq t$, there exists a simulator $\mathcal{S}$ that interacts with $\mathcal{A}$, playing the role of the honest dealer and honest parties $\{P_i\}_{i\in[n]\setminus T}$, such that for any $x_0 \in R$, $View_{\mathcal{A},\mathcal{S}} \equiv_c View_{\mathcal{A},Share}(x_0)$ where $View_{\mathcal{A},\mathcal{S}}$ and $View_{\mathcal{A},\mathsf{Share}}(x_0)$ are the random variables describing the view of $\mathcal{A}$ when interacting with $\mathcal{S}$ and when interacting with the dealer and $\{P_i\}_{i\in[n]\setminus T}$ in the sharing phase* Share *of the VSS, where the dealer has input x_0.*

3 Distributed low-degree proofs

3.1 Interactive (non-ZK) Distributed Low-Degree Proof

We describe now a distributed (interactive) proof for the existence of a low-degree polynomial interpolating a distributed input, under the assumption that no more than t out of n verifiers collude. More precisely, let R be a ring, $\mathcal{E} = \{\alpha_1, \ldots, \alpha_n\} \subseteq R$ an exceptional set and $d \in \mathbb{N}$. Consider the relation

$$\mathcal{R}_{\mathsf{lowdeg}}^{d,\mathcal{E}} = \{(x, f) : x = (x_1, \ldots, x_n) \in R^n,\ f \in R[X]_{d-1},\ f(\alpha_i) = x_i\ \forall i \in [n]\}.$$

Provided each verifier V_i holds x_i, with $x = (x_1, \ldots, x_n)$, our protocol proves x is in the language induced by the relation above. Note we are not concerned with zero-knowledge here, which will be addressed later in Section 3.3.

The proof is based on folding, resembling FRI [BBHR18] and [BFS20]: the problem of showing low-degree is self-reduced at each round to an instance with half the initial degree. Specifically, let $f^{(k-1)}$ be the polynomial claimed to have degree d_{k-1} at the start of round k.

The prover *splits* $f^{(k-1)}$ deterministically into $g_0^{(k)}, g_1^{(k)}$, both polynomials of degree at most $d_k = d_{k-1}/2$. Each verifier V_i receives an evaluation of the two polynomials above, at a certain point which depends on α_i and k.Finally, a random challenge is sampled, and the prover proceeds recursively, showing $f^{(k)} = g_0^{(k)} + \mu g_1^{(k)}$ has degree at most d_k. After τ rounds, $f^{(\tau)}$ is eventually sent in clear. Verifiers can then recursively check the split was correctly computed, relative to their assigned point α_i, using $f^{(\tau)}$ and the evaluations provided by the prover. The proof is eventually accepted if *all* honest verifiers accept it.

As for how to split the polynomials, we consider two possibilities: the first one used in $\Pi^{d,\mathcal{E}}_{\mathsf{lowdeg}}$ (Figure 1) consists on splitting f into high-degree and low degree terms as in [BFS20]. If we can however further assume each of the sets $\mathcal{E}^{(k)} = \{\alpha_i^{2^k} : i \in [n]\}$ for $k \in \{0, \ldots, \tau\}$ to be exceptional (which is the case for fields)[9], we could improve on efficiency setting $f^{(k-1)}(X) = g_0^{(k)}(X^2) + X \cdot g_1^{(k)}(X^2)$ as done in FRI. This is especially beneficial when $\mathcal{E}$ is the set of 2^m-th roots of unity over a field, for some m. This second variant $\Pi^{d,\mathcal{E}}_{\mathsf{lowdeg}-\mathsf{alt}}$, is detailed in Figure 2.

For simplicity, we assume that d is a power of 2, but this can easily be extended to general d, see Section 3.6. For our proof we need a large exceptional set S in the ring R, which may overlap with $\mathcal{E}$. The soundness error will be inversely proportional to the size of S. Soundness amplification techniques are discussed in Section 3.7. The parameter $\tau \leq \log d$ is set for flexibility, so that we stop the recursion when the current polynomial $f^{(\tau)}$ has degree $d/2^\tau$. Nonetheless for asymptotics we will always consider the choice $\tau = \Theta(\log d)$.

Theorem 1. *Protocol $\Pi^{d,\mathcal{E}}_{\mathsf{lowdeg}}$ in Figure 1 is a correct distributed proof for $\mathcal{R}^{d,\mathcal{E}}_{\mathsf{lowdeg}}$ with $1/|S|$-round by round soundness against a malicious prover and up to n corrupted verifiers.*

Proof. Correctness is trivial to verify. Regarding soundness we explicitly describe a *doomed* set D. Let T be the set of corrupted parties, $H = [n] \setminus T$ and $\mathcal{E}_H = \{\alpha_i\}_{i \in H}$. Given a state $(x_i, v_i)_{i \in H}$ until the h-th round, we call $g_0^{(k)}$, $g_1^{(k)}$ the polynomials obtained interpolating the values honest verifiers receive at round k, $f = f^{(0)}$ the interpolation of $(x_i)_{i \in H}$ and $f^{(k-1)} = g_0^{(k)} + X^{d_k} g_1^{(k)}$ for $1 < k \leq \tau$, with μ_k being the k-th round challenge from $\mathcal{F}_{coin}$. Then such tuple of views lies if D if and only if at least one of the following conditions is satisfied:

1. $\deg\left(g_0^{(h)} + \mu_h g_1^{(h)}\right) \geq d_h$.

[9] But it may be a stronger condition on rings; as an example consider $R = \mathbb{Z}_{15}$, where $\mathcal{E} = \{2, 3\}$ is exceptional, while $\mathcal{E}^{(1)} = \{4, 9\}$ is not, since $9 - 4$ divides 0.

Protocol $\Pi^{d,\mathcal{E}}_{\mathsf{lowdeg}}(x, f)$

n-verifier interactive proof for the relation

$$\mathcal{R}^{d,\mathcal{E}}_{\mathsf{lowdeg}} = \{(x, f) \,:\, x \in R^n,\ f \in R[X]_{d-1},\ f(\alpha_i) = x_i\ \forall i \in [n]\}$$

where x is distributed among n verifiers (verifier V_i having as input x_i), and $\mathcal{E} = \{\alpha_1, \ldots, \alpha_n\} \subseteq R$ is an exceptional set. We assume d to be a power of 2 The proof is parametrized by $\tau \in \mathbb{N}$ with $\tau \leq \log_2 d$. For $k \in [\tau]$ we set $d_k = d/2^k$. The proof requires $S \subseteq R$ be a (large) exceptional set, which may overlap with $\mathcal{E}$. $\mathcal{F}^S_{coin}$ is a coin-tossing functionality sampling and broadcasting a random element in S on each invocation.

Interactive Protocol: The prover P sets $f^{(0)} \leftarrow f$. Then for $k \in [\tau]$:

1. P computes $g_0^{(k)}$, $g_1^{(k)}$ of degree $< d_k$ such that $f^{(k-1)} = g_0^{(k)} + X^{d_k} \cdot g_1^{(k)}$
2. P **privately sends** $g_0^{(k)}(\alpha_i)$, $g_1^{(k)}(\alpha_i)$ to each V_i
3. The verifiers call the coin-tossing functionality $\mu_k \leftarrow^{\$} \mathcal{F}^S_{coin}$ with $\mu_k \in S$
4. P computes $f^{(k)} \leftarrow g_0^{(k)} + \mu_k \cdot g_1^{(k)}$

Finally P **broadcasts** the polynomial $f^{(\tau)}$

Verification: Each V_i sets $f^{(k)}(\alpha_i) \leftarrow g_0^{(k+1)}(\alpha_i) + \alpha_i^{d_{k+1}} g_1^{(k+1)}(\alpha_i)$ for $k \in [\tau - 1]$ and accepts if and only if the following checks hold:

5. $\deg f^{(\tau)} < d/2^\tau$
6. $x_i = g_0^{(1)}(\alpha_i) + \alpha_i^{d/2} \cdot g_1^{(1)}(\alpha_i)$
7. $f^{(k)}(\alpha_i) = g_0^{(k)}(\alpha_i) + \mu_k \cdot g_1^{(k)}(\alpha_i)$ for all $k \in [\tau]$

Fig. 1. Distributed low-degree proof $\Pi^{d,\mathcal{E}}_{\mathsf{lowdeg}}$.

2. $f^{(0)}(X) \neq g_0^{(1)}(X) + X^{d/2} \cdot g_1^{(1)}(X)$.
3. $f^{(k)}(X) \neq g_0^{(k)}(X) + \mu_k \cdot g_1^{(k)}(X)$ for some $k \leq h$, $k < \tau$.

If the given input does not lie in the projection of the associated language over indices H, i.e. if there is no polynomial of degree $d-1$ interpolating $(x_i)_{i \in H}$, then we have $(x_i)_{i \in H} \in D$ from the first condition. Next, if a complete transcript lies in D, then conditions 2, 3 implies one verifier rejects, as they are all explicitly checked. Conversely if the complete transcript satisfies condition 1, then either all verifiers reject as $\deg f^{(\tau)} \geq d_\tau$ or $\deg f^{(\tau)} < d_\tau$. The second case however implies $f^{(\tau)}(\alpha_i) \neq g_0^{(\tau)}(\alpha_i) + \mu_\tau g_1^{(\tau)}(\alpha_i)$ for some α_i. This is true as $g_0^{(\tau)} + \mu_\tau g_1^{(\tau)}$ has degree at least d_τ and it is the polynomial of minimum degree taking its values in $\mathcal{E}_H$ since both $g_0^{(\tau)}$ and $g_1^{(\tau)}$ must have degree smaller than $|\mathcal{E}_H| - 1$.

Finally, we show escaping D is hard (condition 2 in Definition 7) Regarding the first message, if $(x_i, \varnothing)_{i \in H} \in D$ then $\deg f^{(0)} \geq d$. Thus for any messages resulting in $g_0^{(1)}, g_1^{(1)}$, if condition 2 is not satisfied, the extended view lies in D. If not instead, at least one of the two polynomials must have degree $\geq d/2$. By Lemma 2 we then have that for a uniformly sampled μ_1 the first condition

is not satisfied with probability $\leq 1/|S|$. For a transcript until the h-th round lying in D instead, if conditions 2 and 3 are satisfied no reply can end outside of D. Conversely if they are both not satisfied, the first one must be. The next message then is either such that

$$g_0^{(h+1)}(X) + X^{d_h/2} g_1^{(h+1)} \neq g_0^{(h)} + \mu_h g_1^{(h)}$$

which implies that the third condition is false, or not, which implies that at least one of $g_0^{(h+1)}, g_1^{(h+1)}$ has degree at least $d_h/2 = d_{h+1}$. Using again Lemma 2, we have that their random linear combination also has degree at least d_{h+1} up to probability $1/|S|$. This conclude our argument for round-by-round soundness.

3.2 Improvements for Specific Rings

Assume now that, for a given exceptional set $\mathcal{E} = \{\alpha_1, \ldots, \alpha_n\}$, all the sets $\mathcal{E}^{(k)} = \{\alpha_i^{2^k} : \alpha_i \in \mathcal{E}\}$ for $k \in [\tau]$ are exceptional and $|\mathcal{E}^{(k)}| = 2 \cdot |\mathcal{E}^{(k+1)}|$. In particular, these assumptions hold if n is a power of 2, R is a finite field $\mathbb{F}$ with a primitive n-th root of unity (i.e. the multiplicative order $|\mathbb{F}| - 1$ is a multiple of n) and $\mathcal{E}$ is the set of all n-th roots of the unity.[10] In these conditions, we present an alternative protocol $\Pi^{d,\mathcal{E}}_{\mathsf{lowdeg\text{-}alt}}$ in Figure 2. The only difference with $\Pi^{d,\mathcal{E}}_{\mathsf{lowdeg}}$ lies in the splitting procedure for f, now divided into even and odd powers terms.

The main advantage of this approach is that, as the domain decreases in size, evaluating the intermediates polynomials becomes faster for the prover. More specifically, P performs $O(n/2^k \cdot \log n)$ ring operations in round k for two FFTs, which in total amounts to $O(n \log n)$. Looking forward, this also improves proof size for the compiled non-interactive proof by a factor 2, see Section 3.4.

3.3 Zero-Knowledge Compiler

We now show how to turn the protocol in Fig. 1 into a distributed zero-knowledge proof. The (standard) idea is to make the prover mask $f(X)$ with a random low-degree polynomial $b(X)$. Specifically, P first shares evaluations of $b(X)$ among the verifiers, and later shows $b(X) + \mu_0 \cdot f(X)$, for a randomly sampled μ_0, to have low degree. The protocol is detailed in Fig. 3.

Theorem 2. *Let Π be a correct distributed proof for $\mathcal{R}^{d,\mathcal{E}}_{\mathsf{lowdeg}}$ with ε-round by round soundness against any number of corruptions. Then $\Pi^{d,\mathcal{E}}_{\mathsf{dZKlowdeg}}$ (Figure 3) is a correct distributed proof for the same relation with $\max(\varepsilon, 1/|S|)$-round by round soundness and perfect zero-knowledge against any number of corruptions.*

We include a proof in Appendix A.1.

Remark 1. Applying either of our distributed proof for low-degree to the protocol in Figure 3 yields a perfect zero-knowledge proof with round-by-round soundness error $1/|S|$.

[10] For instance, the scalar fields of BLS12-377 and BLS12-381 admit 2^m-th roots of unity respectively for all $m \leq 44$ and $m \leq 32$.

Protocol $\Pi^{d,\mathcal{E}}_{\mathsf{lowdeg\text{-}alt}}(x, f)$

Notation as in Figure 1. $\mathcal{E} = \{\alpha_1, \ldots, \alpha_n\}$ and $\mathcal{E}^{(k)} = \{\alpha_1^{2^k}, \ldots, \alpha_n^{2^k}\}$ are exceptional for $k \leq \tau$. Note $\mathcal{E} = \mathcal{E}^{(0)}$. We denote $\alpha_i^{(k)} = \alpha_i^{2^k}$.

Interactive Protocol: As in Figure 1 up to replacing line 1 and 2 in the loop by:

1*. P computes $g_0^{(k)}$, $g_1^{(k)}$ of degree $< d_k$ so that $f^{(k-1)}(X) = g_0^{(k)}(X^2) + X \cdot g_1^{(k)}(X^2)$

2*. P **privately sends** $g_0^{(k)}(\alpha_i^{(k)})$, $g_1^{(k)}(\alpha_i^{(k)})$ to each V_i

Verification: Each V_i sets $f^{(k)}(\alpha_i^{(k)}) \leftarrow g_0^{(k+1)}(\alpha_i^{(k+1)}) + \alpha_i^{(k)} g_1^{(k+1)}(\alpha_i^{(k+1)})$ for $k \in [\tau - 1]$ and accepts iff the checks in Figure 1 hold, up to replacing line 6 with:

6*. $x_i = g_0^{(1)}(\alpha_i^2) + \alpha_i \cdot g_1^{(1)}(\alpha_i^2)$

Fig. 2. Distributed low-degree proof with alternative splitting $\Pi^{d,\mathcal{E}}_{\mathsf{lowdeg\text{-}alt}}$.

Protocol $\Pi^{d,\mathcal{E}}_{\mathsf{dZKlowdeg}}(x, f)$

n-verifier interactive proof for the relation $\mathcal{R}^{d,\mathcal{E}}_{\mathsf{lowdeg}}$. Notation is as in Figure 1. Π is a (non-zk) distributed proof for $\mathcal{R}^{d,\mathcal{E}}_{\mathsf{lowdeg}}$.

Interactive Protocol:

1. P samples $b \leftarrow^{\$} R[X]_{d-1}$ and *privately sends* $b_i \leftarrow b(\alpha_i)$ to V_i
2. The verifiers call the random coin functionality $\mu_0 \leftarrow^{\$} \mathcal{F}^S_{coin}$
3. All parties execute Π: P with input $b(X) + \mu_0 \cdot f(X)$, V_i with input $b(\alpha_i) + \mu_0 \cdot x_i$

Verification: Each verifier V_i accepts if and only if its execution of Π is accepting.

Fig. 3. Zero-knowledge distributed proof of low-degree.

3.4 Removing Interaction in the ROM

In this Section we describe a generic Fiat-Shamir like compiler for distributed proofs in the Random Oracle Model (ROM). This is essentially an adaptation of the one presented in [BCS16] to the distributed setting.

At a high level, let Π be a *public-coin* τ-rounds distributed protocol. That is, one where verifiers can only invoke $\mathcal{F}_{coin}$ at the end of each round, and decide whether to accept or not based on their view after the protocol ends. Given such Π and random oracle $\mathcal{H}$ the non-interactive protocol works as follows. The prover internally simulates Π. At the end of round k, it collects the message M_k it wishes to broadcast and $m_{k,i}$ the message it would have privately sent to V_i. Then it computes c_k as a commitment to the vector of all $m_{k,i}$, $i \in [n]$, and obtain its next challenge μ_k (i.e. the expected output of $\mathcal{F}_{coin}$) as the hash of all the commitments c_j and virtually broadcast message M_j computed so far. When Π halts, P broadcasts all c_k, M_k and privately sends $m_{k,i}$ to V_i along with

opening information. Finally, verifiers accept if all openings are correct and if the transcript is accepting. A detailed description is provided in Figure 4.

FS-like Compiler $\mathsf{FS}[\Pi](x, w)$

$\mathcal{H}$ is a random oracle. MT is a Merkle tree vector commitment realized with $\mathcal{H}$. Π is a τ rounds distributed proof for relation $\mathcal{R}$.

Interactive Protocol: The prover P performs the following:

1. Computes $(c_0, \mathsf{aux}_0) \leftarrow \mathsf{MT.Com}(\mathsf{pp}, x)$ commitment to the distributed input
2. $\mathsf{op}_{0,i} \leftarrow \mathsf{MT.Open}(\mathsf{pp}, i, \mathsf{aux}_0)$ openings of c_0
3. Internally runs Π on input (x, w)
4. **For** $k \in [\tau]$, at the end of the k-th round:
5. Let $m_{k,i}$ be the messages it would have privately sent to V_i
6. Let M_k be the message it would have broadcast
7. $c_k, \mathsf{aux}_k \leftarrow \mathsf{MT.Com}(\mathsf{pp}, m_{k,1}, \ldots, m_{k,n})$, commitment to private messages
8. $\mathsf{op}_{k,i} \leftarrow \mathsf{MT.Open}(\mathsf{pp}, i, \mathsf{aux}_k)$, opening of c_k in position i
9. $\mu_k \leftarrow \mathcal{H}(c_0, c_1, M_1, \ldots, c_k, M_k)$ and set it as the output of $\mathcal{F}_{coin}$ in Π
10. When the execution of Π terminates sets:
11. $\pi \leftarrow (c_0, c_1, M_1, \ldots, c_\tau, M_\tau)$ and $\pi_i \leftarrow (\mathsf{op}_{0,i}, m_{1,i}, \mathsf{op}_{1,i}, \ldots, m_{\tau,i}, \mathsf{op}_{\tau,i})$
12. **Broadcast** π and **privately send** π_i to V_i

Verification: Each verifier V_i retrieves $c_0, \mathsf{op}_{0,i}$ and $M_k, m_{i,k}, c_k, \mathsf{op}_{k,i}$ for $k \in [\tau]$ from (π, π_i). Next it computes μ_k as done in Line 9, and accepts if and only if

13. $1 = \mathsf{MT.Vfy}(\mathsf{pp}, c_0, x_i, i, \mathsf{op}_{0,i})$
14. $1 = \mathsf{MT.Vfy}(\mathsf{pp}, c_k, m_{k,i}, i, \mathsf{op}_{k,i})$ for all $k \in [\tau]$
15. Its local view $(x_i, m_{1,i}, M_1, \mu_1, \ldots, m_{\tau,i}, M_\tau, \mu_\tau)$ is accepting according to Π

Fig. 4. Generic Fiat-Shamir like compiler for distributed proofs.

Theorem 3. *Let Π be a correct τ-rounds distributed proof for $\mathcal{R}$ with ε-round by round soundness against up to t verifiers. Then $\mathsf{FS}[\Pi]$ is correct and ε'-sound for adaptively chosen inputs in the ROM against a q-query adversary $\mathcal{A}$ corrupting up to t verifiers, where*

$$\varepsilon' \;=\; (q+\tau)\varepsilon \;+\; (2q^2)/2^\lambda.$$

In order to preserve zero-knowledge a necessary condition is for the used MT commitment to be hiding (see Section 2.3). Let then $\mathsf{FS}_{\mathsf{zk}}$ be the compiler FS up to replacing MT with hiding MT. This can be shown to preserve zero-knowledge.

Theorem 4. *Let Π be a public-coin zero-knowledge distributed proof for $\mathcal{R}$ against t malicious verifiers. Then $\mathsf{FS}_{\mathsf{zk}}[\Pi]$ is computationally zero-knowledge in the ROM against t malicious verifiers.*

We include the proofs of Theorems 3 and 4 in Appendix A.2.

3.5 Efficiency

Following are the asymptotic costs of protocols in Figure 1-2 made ZK (Figure 3) and then non-interactive via $\mathsf{FS}_{\mathsf{zk}}$ (Figure 4). We set $\tau = \Theta(\log d)$ and assume that the points α_i were pre-assigned to the respective verifier.

Communication. In the general ring setting, the prover broadcasts $\Theta(\log d)$ hash values (the MT roots) and $d\,2^{-\tau} = \Theta(1)$ ring elements. It further privately communicates $\Theta(\log n \cdot \log d)$ hash values and $\Theta(\log d)$ ring elements. For rings supporting the alternative protocol instead, private communication involves only $\leq (\tau+1)\log n - 1/2 \cdot \tau^2$ hashes. For $d \approx n$ those are roughly half as before.

Verifier Computation. Each verifier performs $O(\log d)$ operations[11] in R for checks 6-7 (Figure 1) and $O(\log d \cdot \log n)$ hash evaluations to check the MT openings. As before, the second protocol requires about half hash evaluations for $d \approx n$.

Prover computation. Here the computation is dominated by the evaluations of intermediate polynomials in over $\mathcal{E}$. Without additional assumptions, Hörner's method allows evaluating in $O(nd)$ ring operations[12]. If we can use FFTs then computation is reduced to $O(n \log n \log d)$. Finally, if the alternative protocol is used, this is further reduced to $O(n \log d)$ ring operations as we perform 2 FFTs evaluation on a domain of size $n/2^k$ for a degree $d/2^k$ polynomial, for $0 \leq k \leq \tau$.

In addition to the above, the basic protocol requires $O(n \log n)$ hash evaluations, whereas the alternative one only requires $O(n)$.

3.6 Dealing with any Degree d

The protocols in Figure 1 and Figure 2 can be extended to deal with the case where d is not a power of 2, as follows. Let d_1 be the largest power of two strictly smaller than d, and $d_k = d_1/2^{k-1}$, for $k \geq 1$. Modify then the first loop iteration for $k = 1$ computing $g_0^{(1)}$, $g_1^{(1)}$ such that $f^{(0)}(X) = g_0^{(1)}(X) + X^{d-d_1} g_1^{(1)}(X)$, with the degree conditions $\deg g_0^{(1)}(X) < d - d_1$, $\deg g_1^{(1)}(X) < d_1$ (we use this splitting for the first step even in the case of Figure 2)

From there on all successive steps hold in the same way (with the small change that check 1 by the verifier should be that $\deg f^{(\tau)} < d_1/2^{\tau-1}$). It is easy to see that Theorem 1 would still hold.

3.7 Soundness Amplification

All our interactive constructions have round-by-round soundness error $1/|S|$, where S is the largest exceptional set in R, thus requiring S to be exponentially

[11] Assuming $\alpha_i^{(k)} = \alpha_i^{2^k}$ was precomputed.

[12] Here we are using the fact that each evaluation of a polynomial of degree d_k would take $O(d_k)$ operations in R, and $\sum_{k=0}^{\tau} d_k = \sum_{k=0}^{\tau} d/2^k < 2d$.

large in λ. Note however that there is in principle no guarantee such S exists: indeed, while the relation $\mathcal{R}_{\mathsf{lowdeg}}^{d,\mathcal{E}}$ requires the existence of an exceptional set $\mathcal{E}$, this only needs to be of size n, which is not enough by itself.

Unfortunately, parallel repetitions of (FS-compiled) multi-round proofs may not amplify soundness efficiently [AFK22]. For this reason we show a simple way to improve soundness assuming a ring extension $R \subseteq R'$ with an exponentially large exceptional set $S' \subseteq R'$ is known. The idea is to execute the protocol in Figure 1 replacing R by R' and with S' playing the role of S. However, while this guarantees soundness, it only shows, in principle, the existence of a witness $f \in R'[X]$ but not in $R[X]$. Nevertheless, provided each party also checks $f(\alpha_i) \in R$, this is sufficient as the following lemma shows.

Lemma 3. *Let d, n be positive integers, R a ring with an exceptional set $\mathcal{E} = \{\alpha_1, \ldots, \alpha_n\}$ and extension $R \subseteq R'$ and call $\mathcal{E}_H = \{\alpha_i : i \in H\} \subset \mathcal{E}$ for a given subset $H \subseteq [n]$ of size m. Finally let $x_i \in R$ for all $i \in H$. If there exists a polynomial $f' \in R'[X]_{d-1}$ with $f'(\alpha_i) = x_i$ for all $i \in H$, then there exists a polynomial $f \in R[X]_{d-1}$ with $f(\alpha_i) = x_i$ for $i \in H$. Moreover $d < m \Rightarrow f = f'$.*

Proof. First, by Lemma 1, since $\mathcal{E}_H$ is exceptional and contained in R, we know there is a polynomial $f \in R[X]$ of degree $\leq m-1$ such that $f(\alpha_i) = x_i$ for $i \in H$. If $d \geq m$ we are done. Assume then that $d < m$. Note that f is also in $R'[X]$, hence f and f' are two polynomials in $R'[X]$ with $f(\alpha_i) = f'(\alpha_i)$ for i in H. Since they are both polynomials of degree $\leq m-1$ (because we are in the case $d < m$) then Lemma 1 guarantees $f = f'$ and therefore $f \in R[X]_{d-1}$.

For the simple and commonly encountered case of $R = \mathbb{Z}_{p^k}$ such extension consists of the Galois ring $R' = GR(p^k, r)$ for extension degree $r = \lambda / \log p$. Indeed, it is well known that $GR(p^k, r)$ contains an exceptional set of p^r elements [BCFK21].

4 Verifiable Secret Sharing

4.1 Verifiable Secret Sharing Scheme

In this section we construct a VSS (see Definition 10) starting from the non-interactive distributed proof from the previous section. Our VSS supports Shamir secret sharing of elements in a ring R among n users, as long as there exists an exceptional set $\mathcal{E}^* = \{\alpha_0, \alpha_1, \ldots, \alpha_n\} \subseteq R$ of size at least $n+1$. We also assume a large $S \subseteq R$ exceptional set to instantiate the non-interactive distributed proof in Section 3 with high soundness. Otherwise the techniques presented in Section 3.7 could be used.

At a high level, following [ABCP23], our VSS is obtained combining a (non-interactive) distributed proof of low degree with a complaining phase. The sharing phase consists of three rounds. In the first round, the dealer computes shares x_i of a secret x_0 along with a distributed proof $(\pi, \pi_1, \ldots, \pi_n)$ of low degreeness. It then broadcasts π and privately sends each proof piece to the corresponding

Verifiable Secret Sharing scheme

(t, n)-VSS for sharing a secret $x_0 \in R$. We assume $\mathcal{E}^* = \{\alpha_0, \alpha_1, \ldots, \alpha_n\} \subseteq R$ is an exceptional set, call $\mathcal{E} = \{\alpha_1, \ldots, \alpha_n\}$ and let $S \subseteq R$ be the largest exceptional subset. We call $\Pi = \mathsf{FS}_{\mathsf{zk}}\left[\Pi^{t+1,\mathcal{E}}_{\mathsf{dZKlowdeg}}\right]$. See Sections 3.3-3.4.

Share: Let $x_0 \in R$ be the secret to be shared.

- **First Round**. The dealer proceeds as follows:
  1. Sample a uniformly random $f(X) \leftarrow^{\$} R[X]_t$ such that $f(\alpha_0) = x_0$
  2. Set $x_i := f(\alpha_i)$ the i-th user's share, for $i \in [n]$
  3. Run Π on input (x, f) where $x = (x_i)_{i\in[n]}$ and get $\pi, (\pi_i)_{i\in[n]}$.
  4. **Broadcast** π and **privately send** (x_i, π_i) to V_i
- **Second Round**. Each user P_i, upon receiving x_i and π, π_i, verifies its proof running Π as the i-th verifier. If it rejects, P_i **broadcasts** a complaint bit.
- **Third Round**. For each complaining P_j, the dealer **broadcasts** (x_j, π_j).
- **Finalization**. If all broadcast π_j are accepting, the protocol continues as normal (P_i with share x_i). Otherwise, parties disqualify the dealer and set their share to the default value 0.

Reconstruction: To reconstruct the secret x_0, parties proceed in one round as follows:

- Each party P_i **broadcasts** its local view (x_i, π_i) from the sharing phase
- Each party P_i performs Lagrange interpolation on values x_j with an accepting proof (π, π_j), and returns the evaluation of such polynomial in α_0.

Fig. 5. Verifiable Secret Sharing from.

verifier. Next, in the second round, every user checks the received proof and, if incorrect or missing, broadcasts a complain bit. Finally, in case of complaints, there is a third round where the dealer publicly broadcasts the share and proof of each complaining users, which all parties locally verify.

In order to reconstruct, users simply broadcast their (local) views. If the dealer was previously disqualified[13], then parties agree on 0 being the final secret. Otherwise, each party collects the shares x_j sent with an accepting proof, computes $f(X) \in R[X]_t$ interpolating them, and obtains the secret $x_0 = f(\alpha_0)$. The full protocol is detailed in Figure 5.

Theorem 5. *Suppose $2t + 1 \leq n$ and Π is a non-interactive ZK distributed proof with ε-adaptive soundness, with negligible ε. Then the protocol in Figure 5 is a VSS satisfying correctness, t-privacy and strong commitment in the ROM.*

Proof. We prove each property separately.

[13] This may happen if the dealer addressed a complaint by broadcasting an incorrect proof.

Correctness. Assuming the dealer is honest, let $\mathcal{A}$ be an adversary corrupting T verifiers. Let $\mathcal{B}$ be an adversary playing against the soundness game of the distributed ZKP. Specifically, $\mathcal{B}$ will start the protocol and play the role of the honest parties, until the reconstruction phase, where it picks an index i uniformly at random from the parties in $[n] \setminus H$ that output an accepting view. Then it outputs the partial proof π_i along with the public part of the proof π_{pub}.

Note that the dealer cannot be disqualified at the end of the sharing phase and in case of complaints from corrupt parties it will output views that are going to be accepted by the correctness of the distributed ZKP. Therefore the only step where correctness might fail is in the reconstruction, in the event that a corrupt party opens an incorrect view that passes verification. More specifically, call E the event that at least one party P_i controlled by the adversary $\mathcal{A}$ outputs an incorrect view that verifies in the reconstruction phase. If E happens then $\mathcal{B}$ will pick the corresponding view with probability at least $\frac{1}{t}$. Thus the advantage of $\mathcal{B}$ in winning the soundness game is bounded by $\frac{1}{t}\Pr[\mathsf{E}]$. If the event E does not happen, then all parties output correct views that by the property of correctness of the distributed ZKP are accepted and thus honest parties are able to reconstruct the correct share with probability 1, hence $\mathcal{A}$ does not win the game against the correctness of the VSS. This means that $\text{Adv}^{\text{corr}}_{\text{VSS}}(\mathcal{A}) \leq \Pr[E] \leq t\text{Adv}^{\text{snd}}_{\text{dZKP}}(\mathcal{B})$.

t-Privacy. Given an adversary $\mathcal{A}$ corrupting parties $\{P_i\}_{i\in T}$ for some $T \subseteq [n]$ with $|T| \leq t$, we describe a simulator $\mathcal{S}$ playing the role of the honest parties. First, let $\mathcal{S}_{\Pi}$ be a simulator for Π against an adversary corrupting T parties[14] Then, $\mathcal{S}$ initially samples uniformly randomly $x_i \leftarrow^{\$} R[X]_t$ for $i \in T$. Next, it computes (π, π_i) executing $\mathcal{S}_{\Pi}$ on input $(x_i)_{i\in T}$, and proceed sending (π, x_i, π_i) to corrupted parties. In case of complaints by a corrupted P_i, $\mathcal{S}$ broadcasts its local view (x_i, π_i). For all x_0, we show the resulting transcript to be indistinguishable form one obtained through Share with secret x_0 using a sequence of hybrid distributions.

- D_1: The distribution of corrupted parties' transcripts when $\mathcal{A}$ interacts with $\mathcal{S}$ as above.
- D_2: As D_1 but $\mathcal{S}$ computes $x_i = f(\alpha_i)$ for $f \leftarrow^{\$} R[X]_t$ such that $f(\alpha_0) = x_0$.
- D_3: The distribution of corrupted parties' transcripts when $\mathcal{A}$ interacts with a dealer using Share on input x_0.

D_1 and D_2 follow the same distribution because (by Lemma 1) $|T| \leq t$ implies that $(f(\alpha_i))_{i\in T}$ is a uniformly random vector in $R^{|T|}$ when f is a random polynomial such that $f(\alpha_0) = x_0$. On the other hand, the difference between D_2 and D_3 is that in the former the simulator $\mathcal{S}_{\Pi}$ for Π is used while the latter uses the actual proof Π, but we know the transcripts are indistinguishable by the zero-knowledge property of Π.

[14] In a non-interactive distributed proof, an adversary's behavior is fully determined by the set of parties it corrupts.

Strong Commitment. If the malicious dealer gets disqualified for broadcasting an incorrect proof in the third round, all honest players set their own share to 0. Moreover, in the reconstruction phase they always return 0. Thus in this case strong commitment holds perfectly.

Conversely, if the dealer is not disqualified, each honest user ends Share with share x_i and accepting proof π_i. As we assumed $n \geq 2t+1$, then $H = [n] \setminus T$ has size $|H| \geq t+1$. From ε-soundness we have that, up to probability ε, the shares x_i are the evaluation of a (unique) polynomial $f(X)$ of degree $\leq t$, which identifies a unique secret $x_0 = f(\alpha_0)$.

Next, during the reconstruction phase, let Bad_j for $j \in T$ be the event in which P_j broadcast (x_j, π_j) where π_j is accepting but $f(\alpha_j) \neq x_j$. We then argue $\Pr[\mathsf{Bad}_j] \leq \varepsilon$. Indeed, consider $\mathcal{A}_i$ an adversary for the adaptive-input soundness of Π which corrupts users in $T \setminus \{j\}$. Initially, in runs the t-privacy adversary which return $\pi, (x_i, \pi_i)_{i \in H}$. Next, it correctly simulates an execution of $\mathsf{Reconstruct}$ with the same adversary from which it retrieves (x_j, π_j). If all partial proofs verifier correctly, it returns $(x_i)_{i \in H \cup \{j\}}$ and proof $\pi, (\pi_i)_{i \in H \cup \{j\}}$. If all proofs are accepting but Bad_j occurs, then $\mathcal{A}$ violates adaptive soundness. Thus $\Pr[\mathsf{Bad}_j] \leq \varepsilon$.

Finally, setting $\mathsf{Bad} = \vee_{j \in T} \mathsf{Bad}_j$ we have that $\Pr[\mathsf{Bad}] \leq t\varepsilon$ by a union bound. Assuming $\neg\mathsf{Bad}$ instead, we have that all x_j with an accepting proof are such that $f(\alpha_j) = x_j$. Thus, regardless of the chosen interpolation set, all honest parties recover f during $\mathsf{Reconstruct}$ and in particular they all return $x_0 = f(\alpha_0)$.

4.2 Optimizations

We now detail two optimizations to the VSS scheme presented in the previous section.

Path-pruning. In the third round of the sharing protocol, assume parties P_i for $i \in T$ with $|T| = \vartheta$ issued a complain. Naïvely broadcasting each complaining user's proof would require $O_\eta(\vartheta \log(n)^2)$ communication, with η being the hash output length. This however includes several repeated hash values, as we authenticate every path for every index in T individually. Removing redundant values allows opening a MT of n leaves in $O(\vartheta \log(n/\vartheta))$ hashes (see Section B for a proof). This implies that over arbitrary rings, the complain phase involves the communication of $O(\vartheta \log(n/\vartheta) \log(n))$ hashes (which is in particular $O(n \log n)$). Conversely, over rings where the protocol in Figure 2 can be used, the k-th MT only involves $n \cdot 2^{-k}$ leaves. This implies that with path-pruning communication is bounded by $O(\vartheta \log(n/\vartheta)^2)$ which is in particular $O(n)$.

Two-rounds reconstruction. Since each proof has size $O(\log(n)^2)$, in order to avoid a broadcast complexity of $O(n \log(n)^2)$ we could split the reconstruction phase in two. Initially all parties broadcast their share. Next, they check whether there exists a polynomial of degree t which interpolates those values. This can be checked probabilistically in linear time [CD17]. Finally, all parties broadcast their local proofs as in the previous protocol.

In spite of the possible fall back to an $O(n\log(n)^2)$ sized broadcast, this only occurs when at least one *actively* malicious user is identified. Such case could then be mitigated through incentives/stake, although analyzing such option in detail is outside of our scope.

4.3 Efficiency

In light of the optimizations previously discussed, we now detail the overall sharing costs of our VSS. We call η the hash output length, $\rho = \log_2 |R|$. In the optimistic case in which no complaint is raised we have:

- *Communication.* The dealer needs to broadcast $O_\eta(\log(n)) + O_\rho(1)$ bits and privately send $O_\eta(\log(n)^2) + O_\rho(\log(n))$ bits to each user.
- *Dealer computation.* This is dominated by the proof cost, which amounts to $O(n\log(n)^2)$ and $O(n\log n)$ ring operations, when using respectively the protocol in Figure 1 or Figure 2.
- *Parties computation.* Each party needs to do $O(\log(n))$ ring operations and $O(\log(n)^2)$ hash evaluations to check the openings of the commitments they received.

Next, we study the overhead induced by ϑ complains in the sharing phase. We remark verification time remains sub-linear as long as $\vartheta = o(n/\log(n)^2)$.

- *Communication.* Using *path-pruning*, Protocol 1 requires $O_\mu(\vartheta\log(n/\vartheta)\log n) + O_\rho(\vartheta)$ extra broadcast communication, there the first term is $O_\mu(n\log n)$ in the worst case. Conversely, Protocol 2 requires $O_\mu(\vartheta\log(n/\vartheta)^2) + O_\rho(\vartheta)$ additional bits, where the first term is $O_\mu(n)$ in the worst case.
- *Dealer computation.* The dealer costs remains the same asymptotically.
- *Parties computation.* Each party needs to do extra $O(\vartheta\log(n/\vartheta)\log(n))$ hash evaluations (respectively $O(\vartheta\log(n/\vartheta)^2)$ using protocol 2) and $O(\vartheta)$ ring operations.

Acknowledgements

The authors would like to thank Dario Fiore and Thomas Attema for providing helpful comments.

This work is supported by Juan de la Cierva grant JDC2022-049711-I, the PRODIGY Project (TED2021-132464B-I00) and the ESPADA project (PID2022-142290OB-I00), all funded by MCIN/AEI/10.13039/501100011033.The Juan de la Cierva grant and the PRODIGY project are cofunded by the European Union «NextGenerationEU/PRTR»; ESPADA is cofunded by FEDER, UE. This work has also been supported by the PICOCRYPT project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 101001283);

and by the CONFIDENTIAL6G funded by the European Union (GA 101096435). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the European Commission can be held responsible for them.

References

ABCP23. Shahla Atapoor, Karim Baghery, Daniele Cozzo, and Robi Pedersen. VSS from distributed ZK proofs and applications. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology - ASIACRYPT 2023 - 29th International Conference on the Theory and Application of Cryptology and Information Security, Guangzhou, China, December 4-8, 2023, Proceedings, Part I*, volume 14438 of *Lecture Notes in Computer Science*, pages 405–440. Springer, 2023.

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, Heidelberg, August 2020.

AFK22. Thomas Attema, Serge Fehr, and Michael Klooß. Fiat-shamir transformation of multi-round interactive proofs. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 113–142. Springer, Heidelberg, November 2022.

AJM+23. Ittai Abraham, Philipp Jovanovic, Mary Maller, Sarah Meiklejohn, and Gilad Stern. Bingo: Adaptivity and asynchrony in verifiable secret sharing and distributed key generation. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part I*, volume 14081 of *LNCS*, pages 39–70. Springer, Heidelberg, August 2023.

AKP20. Benny Applebaum, Eliran Kachlon, and Arpita Patra. The round complexity of perfect MPC with active security and optimal resiliency. In *61st FOCS*, pages 1277–1284. IEEE Computer Society Press, November 2020.

Bag23. Karim Baghery. Π: A unified framework for verifiable secret sharing. *IACR Cryptol. ePrint Arch.*, page 1669, 2023.

BBC+19. Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Zero-knowledge proofs on secret-shared data via fully linear PCPs. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 67–97. Springer, Heidelberg, August 2019.

BBHR18. Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *ICALP 2018*, volume 107 of *LIPIcs*, pages 14:1–14:17. Schloss Dagstuhl, July 2018.

BCFK21. Alexandre Bois, Ignacio Cascudo, Dario Fiore, and Dongwoo Kim. Flexible and efficient verifiable computation on encrypted data. In Juan Garay, editor, *PKC 2021, Part II*, volume 12711 of *LNCS*, pages 528–558. Springer, Heidelberg, May 2021.

BCR+19. Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct arguments for R1CS. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 103–128. Springer, Heidelberg, May 2019.

BCS16. Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 31–60. Springer, Heidelberg, October / November 2016.

BFS20. Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent SNARKs from DARK compilers. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 677–706. Springer, Heidelberg, May 2020.

BGW88. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In Janos Simon, editor, *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 1–10. ACM, 1988.

BKP11. Michael Backes, Aniket Kate, and Arpita Patra. Computational verifiable secret sharing revisited. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 590–609. Springer, Heidelberg, December 2011.

CCH+18. Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, and Ron D. Rothblum. Fiat-Shamir from simpler assumptions. Cryptology ePrint Archive, Report 2018/1004, 2018. `https://eprint.iacr.org/2018/1004`.

CCP22. Anirudh Chandramouli, Ashish Choudhury, and Arpita Patra. A survey on perfectly secure verifiable secret-sharing. *ACM Comput. Surv.*, 54(11s):232:1–232:36, 2022.

CD17. Ignacio Cascudo and Bernardo David. SCRAPE: Scalable randomness attested by public entities. In Dieter Gollmann, Atsuko Miyaji, and Hiroaki Kikuchi, editors, *ACNS 17*, volume 10355 of *LNCS*, pages 537–556. Springer, Heidelberg, July 2017.

CD20. Ignacio Cascudo and Bernardo David. ALBATROSS: Publicly AttestabLe BATched Randomness based On Secret Sharing. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part III*, volume 12493 of *LNCS*, pages 311–341. Springer, Heidelberg, December 2020.

CD23. Ignacio Cascudo and Bernardo David. Publicly verifiable secret sharing over class groups and applications to DKG and YOSO. *To appear at Eurocrypt 24. IACR Cryptol. ePrint Arch.*, page 1651, 2023.

CDGK22. Ignacio Cascudo, Bernardo David, Lydia Garms, and Anders Konring. YOLO YOSO: Fast and simple encryption and secret sharing in the YOSO model. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 651–680. Springer, Heidelberg, December 2022.

CDN15. Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.

CG22. Ignacio Cascudo and Emanuele Giunta. On interactive oracle proofs for boolean R1CS statements. In Ittay Eyal and Juan A. Garay, editors, *FC 2022*, volume 13411 of *LNCS*, pages 230–247. Springer, Heidelberg, May 2022.

CP23. Ashish Choudhury and Arpita Patra. On the communication efficiency of statistically secure asynchronous MPC with optimal resilience. *J. Cryptol.*, 36(2):13, 2023.

Fel87. Paul Feldman. A practical scheme for non-interactive verifiable secret sharing. In *28th Annual Symposium on Foundations of Computer Science, Los Angeles, California, USA, 27-29 October 1987*, pages 427–437. IEEE Computer Society, 1987.

GHL22. Craig Gentry, Shai Halevi, and Vadim Lyubashevsky. Practical non-interactive publicly verifiable secret sharing with thousands of parties. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part I*, volume 13275 of *LNCS*, pages 458–487. Springer, Heidelberg, May / June 2022.

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

GJKR07. Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Secure distributed key generation for discrete-log based cryptosystems. *Journal of Cryptology*, 20(1):51–83, January 2007.

GRR98. Rosario Gennaro, Michael O. Rabin, and Tal Rabin. Simplified VSS and fast-track multiparty computations with applications to threshold cryptography. In Brian A. Coan and Yehuda Afek, editors, *Proceedings of the Seventeenth Annual ACM Symposium on Principles of Distributed Computing, PODC '98, Puerto Vallarta, Mexico, June 28 - July 2, 1998*, pages 101–111. ACM, 1998.

KMM+23. Aniket Kate, Easwar Vivek Mangipudi, Pratyay Mukherjee, Hamza Saleem, and Sri Aravinda Krishnan Thyagarajan. Non-interactive VSS using class groups and application to DKG. *IACR Cryptol. ePrint Arch.*, page 451, 2023.

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, Heidelberg, August 1992.

PSL80. Marshall Pease, Robert Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *Journal of the ACM (JACM)*, 27(2):228–234, 1980.

Sch99. Berry Schoenmakers. A simple publicly verifiable secret sharing scheme and its application to electronic. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 148–164. Springer, Heidelberg, August 1999.

Sha79. Adi Shamir. How to share a secret. *Communications of the Association for Computing Machinery*, 22(11):612–613, November 1979.

SS23. Victor Shoup and Nigel P. Smart. Lightweight asynchronous verifiable secret sharing with optimal resilience. *IACR Cryptol. ePrint Arch.*, page 536, 2023.

A Postponed Proofs

A.1 Adding Zero-Knowledge

Proof of Theorem 2. Correctness follows from Π. Next we show soundness and zero-knowledge.

Round-by-Round Soundness. Let D be a set of *doomed* states for Π realizing ε-round-by-round soundness. Then define D' set of states for the protocol in Figure 3 such that, for any set H of non corrupted parties (with associated points $\mathcal{E}_H = \{\alpha_i\}_{i\in H}$)

1. $(x_i, \varnothing)_{i\in H} \in D'$ iff f interpolating x_i over $\mathcal{E}_H$ has $\deg f \geq d$.
2. $(x_i, b_i, \mu_0, v_i)_{i\in H} \in D'$ iff $(b_i + \mu_0 x_i, v_i)_{i\in H} \in D$, where v_i could be empty.

Is easy to verify that an input not lying in the projection of the associated language of $\mathcal{R}^{d,\mathcal{E}}_{\mathsf{lowdeg}}$ satisfies the condition 1. Similarly, a complete state in D' is by condition 2 such that its sub-proof of low-degree for $b + \mu_0 f$ lies in the doomed set D. Thus by Definition 7 Π rejects and so does our protocol for some verifier.

Finally we show escaping D' is hard. Indeed, by Lemma 2, if $\deg f(X) \geq d$, regardless of the degree of $b(X)$, with $b(X)$ being the interpolation of b_i over $\mathcal{E}_H$ of $(b_i)_{i\in H}$, their random linear combination has degree at least d with probability $1 - 1/|S|$. In such case moreover $(b_i + \mu_0 x_i)_{i\in H}$ does not lie in the projection of the associated language $\mathcal{R}^{d,\mathcal{E}}_{\mathsf{lowdeg}}$. Thus $(x_i, b_i, \mu_0)_{i\in H} \in D$. Conversely, non-initial states lie in D' only if condition 2 is satisfied, and in particular only if the sub-view lies in D. Thus the probability it does not lie in D' upon adding the next message and challenge is smaller than ε.

Strong Zero-Knowledge. Let T be the set of malicious verifiers. The simulator Sim takes as input the shares $(x_i)_{i\in T}$ corresponding to the dishonest parties. First, it chooses a polynomial $f^*(X)$ of degree $< d$ such that $f(\alpha_i) = x_i$ for $i \in T$. This can be done because the distributed input lies in the associated language of $\mathcal{R}^{d,\mathcal{E}}_{\mathsf{lowdeg}}$. Next it samples $\mu_0 \leftarrow^{\$} S$, the output of $\mathcal{F}^S_{coin}$, and $r(X) \leftarrow^{\$} R[x]_{d-1}$. Finally it sets $b(X) = r(X) - \mu_0 f(X)$, and proceeds correctly executing Π giving r on input to the prover. Eventually Sim returns $(b(\alpha_i))_{i\in T}$, μ_0 and the view of each malicious verifier during the execution of Π.

The output of Sim follows the same distribution of the adversary's view in the real protocol. Indeed, as r is uniform, $b = r - \mu_0 f^*$ is uniformly distributed over $R[X]_{d-1}$, as in the real protocol. Moreover, conditioning on b, in both worlds the execution of Π is performed with a prover holding r that is a uniformly distributed polynomial satisfying $r(\alpha_i) = b(\alpha_i) + \mu_0 x_i$ for all $i \in T$. Thus the views of malicious parties executing Π follows the same distribution conditioning on μ_0 and b This concludes the proof.

A.2 Non-Interactive Case

Proof of Theorem 3. Correctness readily follows from that of Σ. Regarding soundness, let T be the set of corrupted parties indices and $H = [n] \setminus T$, and $\mathcal{A}$ an adversary corrupting parties in T. At the end of $\mathcal{A}$'s execution, let $z_1, \ldots, z_q$ be the values it queries to $\mathcal{H}$. Calling Coll the event that two distinct z_i, z_j collide, then $\Pr[\mathsf{Coll}] \leq q^2/2^\lambda$ assuming $\mathcal{H}$ outputs λ bits. Next let π and $(x_i, \pi_i)_{i\in H}$ the values returned by $\mathcal{A}$, where $\pi = (c_0, c_1, M_1, \ldots, c_\tau, M_\tau)$ and $\pi_i = (\mathsf{op}_{0,i},$

$m_{1,i}, \mathsf{op}_{1,i}, \ldots, m_{\tau,i}, \mathsf{op}_{\tau,i}$). Up to increasing the number of queries by τ, we can assume $\mathcal{A}$ queried before halting $\mu_k \leftarrow \mathcal{H}(c_0, c_1, M_1, \ldots, c_k, M_k)$. Moreover we assume without loss of generality $\mathcal{A}$ does not repeat queries.

Next we argue $\mathcal{A}$ must know an opening for c_k for all honest parties' indices before computing μ_k. Let BadMT_k the event in which, given all previous RO queries before computing μ_k, an opening of c_k to position i is not known. Assuming $\neg\mathsf{Coll}$ we show this to happen with negligible probability if (π, π_i) is accepting. Indeed, let $h_1, \ldots, h_\ell, h^*_\ell$ be the longest co-path for position i that can be observed among $\mathcal{A}$'s queries before obtaining μ_k. I.e. such that (up to adjusting the order of $\mathcal{H}$'s inputs to match the bit representation of i)

$$h^*_{\ell-1} = \mathcal{H}(h^*_\ell \,||\, h_\ell), \quad \ldots \quad h^*_1 = \mathcal{H}(h^*_2 \,||\, h_2), \quad c_k = \mathcal{H}(h^*_1 \,||\, h_1)$$

Note this is unique thanks to $\neg\mathsf{Coll}$. If the proof is accepting, $\mathcal{A}$ eventually provides a full path to $m_{i,k}$ and in particular finds a preimage for h^*_ℓ, which occurs only with probability $q \cdot 2^{-\lambda}$. Let $\mathsf{BadMT} = \mathsf{BadMT}_0 \vee \ldots \vee \mathsf{BadMT}_\tau$ and Accept the event in which $\pi, (x_i, \pi_i)_{i \in H}$ is accepted by all verifier. With a union bound we obtain

$$\Pr\left[\mathsf{Accept},\ \mathsf{BadMT},\ \neg\mathsf{Coll}\right] \ \leq\ \frac{q(\tau+1)}{2^\lambda} \ \leq\ \frac{q^2}{2^\lambda}.$$

Finally we study the probability of Accept when $(x_i)_{i \in H}$ cannot be extended to an input in the language. By ε-round by round soundness there exists a set D of *doomed* views satisfying Definition 7. Then we define Escape_j the event in which the j-th RO query produces a state not in D from one in D. Formally Escape_j is the event in which for some index $h \in [\tau]$

1. The j-th query is $\mu'_h = \mathcal{H}(c'_0, c'_1, M'_1, \ldots, c'_h, M'_h)$.
2. c'_k is a MT s.t. the first j RO queries define an opening to $m'_{k,i}$ in position $i \in H$, for all $k \in [h]$.
3. $\mu'_k = \mathcal{H}(c'_0, c'_1, M'_1, \ldots, c'_k, M'_k)$ for $k \in [h]$.
4. $v_i = (m'_{1,i}, M'_1, \mu'_1, \ldots, m'_{h-1,i}, M'_{h-1}, \mu'_{h-1})$ the view of V_i until round $h-1$.
5. $(x_i, v_i)_{i \in H} \in D$.
6. $(x_i, v_i, m'_{h,i}, M'_h, \mu'_h) \notin D$, i.e. escapes D at round h.

Note that if $\neg\mathsf{Coll}$ all those values are unique. Moreover, by condition 1 and 2, μ'_k is sampled uniformly and independently from all messages $m'_{k,i}, M'_i$, as they figure in previous RO queries. Thus, the probability that the extended views lie outside of D (conditions 5-6) is smaller that ε.

$$\Pr\left[\mathsf{Escape}_j,\ \neg\mathsf{Coll}\right] \ \leq\ \varepsilon. \quad \Rightarrow \quad \Pr\left[\mathsf{Escape},\ \neg\mathsf{Coll}\right] \leq q \cdot \varepsilon$$

where Escape is the disjunction of all Escape_j for $j \in [q]$. Combining the three bounds obtained so far

$$\begin{aligned}\Pr[\mathsf{Accept}] &\leq \Pr[\mathsf{Accept},\ \neg\mathsf{Coll}] + \frac{q}{2^\lambda} \\ &\leq \Pr[\mathsf{Accept},\ \neg\mathsf{BadMT},\ \neg\mathsf{Coll}] + \frac{2q^2}{2^\lambda} \\ &\leq \Pr[\mathsf{Accept},\ \neg\mathsf{Escape},\ \neg\mathsf{BadMT},\ \neg\mathsf{Coll}] + q \cdot \varepsilon + \frac{2q^2}{2^\lambda}.\end{aligned}$$

To conclude we show the first term in the last expression to be 0. We do so showing by induction that, calling $v_{k,i} = (m_{1,i}, M_1, \mu_1, \ldots, m_{k,i}, M_k, \mu_k)$, we have $(x_i, v_{k,i}) \in D$. The base case $(x_i, \varnothing)_{i \in H} \in D$ follows by Definition 7 as the distributed input chosen by $\mathcal{A}$ cannot lie in the projection of the associated language. Next, assuming the thesis for k, we have that $(x_i, v_{k,i}) \in D$ and, as we assumed $\neg\mathsf{BadMT}$, the condition 2 is satisfied. Therefore, the assumption $\neg\mathsf{Escape}$ implies that 6 must be false and in particular $(x_i, v_{k+1,i})_{i \in H} \in D$ We thus conclude the list of final transcripts to be in D which, by Definition 7 implies that at least one verifier rejects. The thesis follows.

Proof of Theorem 4. For the sake of presentation we proceed assuming MT is a hiding VC [Giu23], which readily follows as $\mathcal{H}(x, r)$ is a computationally hiding commitment to x.

Let $\mathcal{A}$ an adversary for $\mathsf{FS}_{\mathsf{zk}}[\Pi]$ corrupting parties in $T \subseteq [n]$. Because Π is zero-knowledge, there exists a simulator $\mathcal{S}$ against an adversary which corrupts all verifiers in the same set T, but which during the execution of the interactive protocol behaves honestly.

We can then define $\mathcal{T}$ simulator for $\mathcal{A}$. Initially it executes $\mathcal{S}$ to get the simulated views $(x_i, v_i)_{i \in T}$ of each party. It then parses $v_i = (m_{1,i}, M_1, \mu_1, \ldots, m_{\tau,i}, M_\tau, \mu_\tau)$. Next, it generates the vectors $\mathbf{x}, \mathbf{m}_k$ setting them to 0 in position associated to honest users, and $x_i, m_{k,i}$ for malicious ones, i.e. for $i \in T$. With them it computes c_0, c_k hiding-MT commitments to $\mathbf{x}, \mathbf{m}_k$ and $\mathsf{op}_{k,i}$ their respective openings. Finally it programs $\mathcal{H}(c_0, c_1, M_1, \ldots, c_k, M_k) = \mu_k$ and sets

$$\pi = (c_0, c_1, M_1, \ldots, c_\tau, M_\tau), \qquad \pi_i = (\mathsf{op}_{0,i}, m_{1,i}, \mathsf{op}_{1,i}, \ldots, m_{\tau,i}, \mathsf{op}_{\tau,i}).$$

To conclude we proceed through an hybrid argument for distributions D_1, D_2, D_3 where:

D_1: The output distribution of $\mathcal{T}$.

D_2: The output of $\mathcal{T}$ replacing $(x_i, v_i)_{i \in T} = View_{\Pi^\star, T}(x, w)$.

D_3: The view with an honest prover $View_{\mathsf{FS}_{\mathsf{zk}}[\Pi]^\star, T}(x, w)$.

Distinguishing D_1, D_2 reduces to distinguishing between the simulator $\mathcal{S}$'s output and the honest view of its adversary. Regarding D_2, D_3 instead, their only differences are the values used in $c_0, \ldots, c_\tau$ for honest indices, i.e. $i \notin T$. However, as no opening for those positions is provided in both distributions, the two are indistinguishable (through a standard sequence of $\tau + 1$ hybrids) by the hiding property of the VC in use. This concludes the proof.

B Merkle Trees

In this section we discuss the opening size of a Merkle-tree when several positions are opened at once (see for instance [BCR+19, CG22]). Specifically, we assume c to be a MT commitment to n leaves, with n power of 2, and ask how many hash values are necessary to open positions $S \subseteq [n]$ with $|S| = \vartheta$. Eventually we upper-bound the opening size with $\vartheta \lceil \log n/\vartheta \rceil$ hash values, along with the content of those ϑ leaves, in the worst case. As a stepping stone, a preliminary lower bound is presented in the following Lemma, only mildly improving on the trivial one $\vartheta \log n$. There, we denote Sign the sign function, which is -1 for negative input, 0 on input 0, and 1 for positive values.

Lemma 4. *Given a MT-commitment to n leaves, n a power of 2, and calling π a minimal opening for positions $S \subseteq [n]$ with $|S| = \vartheta$, then*

$$|\pi| \cdot \eta^{-1} \;\leq\; \vartheta(\log(n) - 1) + \mathsf{Sign}(\vartheta)$$

with η being the hash function output length.

Proof. If $\vartheta = 0$, no hash has to be sent, and the above expression evaluates to 0. If $\vartheta = 1$, the cost is $\log(n)$ hashes regardless of the position. Finally, for $\vartheta \geq 1$, given a co-path for the first element in S, we can compute both left and right preimage of the root. Hence, openings for the remaining elements in S are either opening of the left or right sub-tree, both of which have $n/2$ leaves. Thus those $\vartheta - 1$ can be opened with $\log n - 1$ hashes and in conclusion

$$|\pi| \cdot \eta^{-1} \;\leq\; \vartheta + (\vartheta - 1)(\log(n) - 1) \;=\; \vartheta(\log(n) - 1) + \mathsf{Sign}(\vartheta).$$

Next, using this Lemma, we will show a tighter bound on the opening size.

Theorem 6. *Given a MT-commitment to n leaves, n a power of 2, and calling π a minimal opening for positions $S \subseteq [n]$ with $|S| = \vartheta$, then*

$$|\pi| \cdot \eta^{-1} \;\leq\; \vartheta \left\lceil \log(n/\vartheta) \right\rceil .$$

with η being the hash output length.

Proof. The proofs works dividing the Merkle-tree into α sub-trees each with n/α leaves, where α is a power of two. Next, to provide an upper-bound to the opening cost, we assume all those α sub-trees as opened individually. Note that for those sub-trees containing a leaf in S, the root does not need to be given in the opening, as it can be computed that element co-path in the sub-tree. Conversely, the root has to be added to the proof. Thus, calling ϑ_i the number of leaves S contains in the i-th sub-tree, for $i \in [\alpha]$, we have that the cost of opening such i-th sub-tree is

$$\begin{aligned} c_i \cdot \eta^{-1} \;&\leq\; \vartheta_i(\log(n/\alpha) - 1) + \mathsf{Sign}(\vartheta_i) + (1 - \vartheta_i) \\ &=\; \vartheta_i \log(n/\alpha) - \vartheta_i + 1. \end{aligned}$$

The first term in the above expression follows from Lemma 4, while the term $(1 - \mathsf{Sign}\,\vartheta_i)$ accounts for the root of the sub-tree, given only if $\vartheta_i = 0$. Finally, the total opening cost is smaller than the sum of those terms:

$$\begin{aligned} |\pi| \cdot \eta^{-1} \;&\leq\; \sum_{i=1}^{\alpha} c_i \cdot \eta^{-1} \;\leq\; \sum_{i=1}^{\alpha} \vartheta_i \log \beta - \vartheta + 1 \\ &=\; \vartheta \log(n/\alpha) - \vartheta + \alpha. \end{aligned}$$

Finally setting α as the largest power of 2 such that $\alpha \leq \vartheta$, we get that $\log(n/\alpha) = \lceil \log(n/\vartheta) \rceil$ and in particular $|\pi| \cdot \eta^{-1} \leq \vartheta \lceil \log(n/\vartheta) \rceil$.