On the Security of KZG Commitment for VSS

Atsuki Momose
University of Illinois at Urbana-Champaign
Urbana, IL, USA
atsuki.momose@gmail.com

Sourav Das
University of Illinois at Urbana-Champaign
Urbana, IL, USA
souravd2@illinois.edu

Ling Ren
University of Illinois at Urbana-Champaign
Urbana, IL, USA
renling@illinois.edu

ABSTRACT

The constant-sized polynomial commitment scheme by Kate, Zaverucha, and Goldberg (Asiscrypt 2010), also known as the KZG commitment, is an essential component in designing bandwidth-efficient verifiable secret-sharing (VSS) protocols. We point out, however, that the KZG commitment is missing two important properties that are crucial for VSS protocols.

First, the KZG commitment has not been proven to be *degree binding* in the standard adversary model without idealized group assumptions. In other words, the committed polynomial is not guaranteed to have the claimed degree, which is supposed to be the reconstruction threshold of VSS. Without this property, shareholders in VSS may end up reconstructing different secrets depending on which shares are used.

Second, the KZG commitment does not support polynomials with different degrees at once with a single setup. If the reconstruction threshold of the underlying VSS protocol changes, the protocol must redo the setup, which involves an expensive multi-party computation known as the powers of tau setup.

In this work, we augment the KZG commitment to address both of these limitations. Our scheme is degree-binding in the standard model under the strong Diffie-Hellman (SDH) assumption. It supports any degree $0 < d \leq m$ under a powers-of-tau common reference string with $m+1$ group elements generated by a one-time setup.

CCS CONCEPTS

• **Security and privacy** → **Distributed systems security**;

KEYWORDS

Verifiable Secret-Sharing; KZG Polynomial Commitment

1 INTRODUCTION

Verifiable secret-sharing (VSS) [15] allows a designated dealer to share a secret with a set of n nodes, of which at most f nodes are malicious. Later, the set of nodes runs an interactive protocol to recover the secret. VSS has been used as a crucial component to design secure protocols of multi-party computation (MPC) [6, 31], distributed key generation [19, 24], randomness beacon [8, 18], and many more. MPC protocols, in particular, require the dealer to simultaneously share multiple secrets, often proportional to the circuit size. We refer to VSS where the dealer shares multiple secrets simultaneously as a *multi-secret* VSS.

Existing efficient constructions of VSS, especially multi-secret VSS [2, 41] crucially rely on polynomial commitment schemes with constant commitment size and constant evaluation proof. Concretely, they use the celebrated Kate-Zaverucha-Goldberg polynomial commitment scheme [27], here on referred to as the KZG commitment.

Polynomial commitment, when used for VSS, must be *degree binding*, i.e., the degree of the polynomial used to share the secret must match the threshold f (or lower). Intuitively, this ensures that each honest node outputs the same secret during the recovery protocol. Otherwise, interpolations using different sets of $f+1$ shares would result in honest nodes outputting different secrets.

Despite its established use, this important degree-binding property has not been proven for the KZG commitment. The original KZG paper [27, 28] proves a similar property called *strong correctness* using a non-standard polynomial Diffie-Helman (PDH) assumption. We observe that the strong correctness property is insufficient for VSS (we will elaborate on this in §3). Very recently, Abraham et al. [2] prove the degree binding property, which they refer to as *interpolation binding*, assuming hardness of Strong Diffie Helman (SDH) [9] in the Algebraic Group Model [23], which significantly constrains the adversary's capabilities. This strong assumption has to do with the common reference string (CRS) of the KZG commitment. Specifically, the KZG commitment to a degree-f polynomial $\phi(\cdot)$ is represented as $g^{\phi(\tau)}$ for generator $g \in \mathbb{G}$ and a trapdoor $\tau \in \mathbb{Z}_q^*$, which is computed from a powers-of-tau CRS $[g, g^{\tau}, \ldots, g^{\tau^f}]$. Obviously, the degree binding property of the KZG commitment relies on the fact that the adversary does not know $g^{\tau^{f+k}}$ for any $k > 0$. However, it is hard to argue that the adversary cannot compute the higher powers $g^{\tau^{f+k}}$ in the standard model where the adversary can access the group representation and perform arbitrary operations on them. This is what leads Abraham et al. to resort to an idealized group model (such as AGM) to prove the degree binding property of the KZG commitment.

Relying on the incapability of computing higher powers for degree-binding introduces another issue. VSS protocols built on the original KZG commitment are not *reconfiguration friendly*. Namely, the protocols are unsuitable for systems in which the number of participants and the fault threshold changes dynamically [2, 41]. In reconfigurable systems where the threshold f changes during the execution, the protocol must use a separate CRS with a distinct trapdoor for each threshold. This is undesirable as the size of the CRS may become prohibitively large.

Our result. The main contribution of this work is to augment the KZG commitment to make it degree-binding against the standard adversary, i.e., without idealized group assumptions. Our augmented KZG commitment supports any degree $0 < d \leq m$ with a

single CRS $[g, g^{\tau}, \ldots, g^{\tau^m}]$, and hence efficiently supports reconfigurable systems. Building on our augmented KZG commitment, we design a multi-secret VSS with optimal communication. Formally,

Theorem 1.1. *Assuming the existence of a public-key infrastructure, random oracle, and a universal common reference string under the SDH assumption, there exists a multi-secret VSS protocol with $O(\kappa Ln + \kappa n^2)$ communication tolerating $f < n/3$ corruption, where L is the number of secrets and κ is the security parameter.*

The core technical ingredient is the proof of degree from an aggregated linear-sized commitment, such as Feldman commitment [21]. Our key idea is that, instead of making sure an adversary is incapable of committing to a higher-degree polynomial, we make it detectable. Specifically, we allow an adversary to compute a KZG commitment to a higher-degree polynomial, but its degree is revealed due to the standard degree-binding property of the Feldman commitment. Since we do not rely on the adversary's inability to compute higher powers, we do not have to assume an idealized group model (like [2] does). The Feldman commitment is of $O(\kappa n)$ size and hence cannot be used directly. We instead aggregate $L = O(n)$ commitments to amortize the cost.

Not relying on the inability to compute higher powers in CRS makes our scheme reconfiguration-friendly. This has an immediate impact on existing dynamic committee threshold cryptography. For example, the state-of-the-art dynamic committee proactive secret-sharing (DPSS) [26, 42] adopts the original KZG commitment to batch-amortize the communication cost. These protocols require a powers-of-tau setup every time the committee is resized. These repeated setups can be avoided if our augmented KZG is used instead. We also present a DPSS protocol using our VSS protocol in Appendix C.

Remark on proof size. Here, we reiterate that our augmented KZG commitment is always linear-sized, so it may not be suitable for a single polynomial. However, we also note that in the context of VSS, the KZG commitment is specifically useful when dealing with a linear number of polynomials [3, 41]. For a single-secret VSS involving a single polynomial, standard linear-sized commitments such as Feldman or Pedersen commitments are sufficient since we already incur quadratic communication in other aspects.

Remark on the network model. For ease of exposition, we present our multi-secret VSS protocol assuming a synchronous network. Our VSS protocol can be easily extended to tolerate asynchronous networks using existing techniques (cf. §5.5). Our DPSS protocol, however, works only in the synchrony model.

Organization. The rest of the paper is organized as follows. After providing the model, the problem definitions, and some preliminaries in §2, we give an overview of the key technical highlights in §3. We present our augmented KZG commitment in §4 and our multi-secret VSS protocol in § 5. We review related works in §6 and conclude with discussions in §7.

2 MODEL AND PRELIMINARIES

We consider a system of n nodes (numbered from 1 to n) of which at most $f < n/3$ are corrupt. All corrupt nodes are controlled by a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$. The adversary chooses which nodes to corrupt upfront, i.e., we assume *static* corruption. Also, we assume that every pair of nodes can communicate over an *authenticated* and *private* channel, which is commonly implemented with a digital signature and symmetric/asymmetric encryption under a public-key infrastructure.

When we assume the synchrony model, we consider a simple lockstep round model. Any message sent by an honest node within a round is delivered to the recipient by the end of that round. Unless explicitly stated otherwise, a value in this paper is an element of a prime field $\mathbb{Z}_q$ where $q \geq 2^{\kappa}$ and a polynomial is an element of $\mathbb{Z}_q[x]$. Let g be a generator of a group $\mathbb{G}$ of order q such that a bilinear pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ exists [10]. We use $[a]$ to denote an ordered set $\{1, \ldots, a\}$ and use $[a, b]$ to denote $\{a, \ldots, b\}$. We use the bold notation $\boldsymbol{x}$ to denote a vector. For a vector of polynomials $\boldsymbol{\phi} = [\phi_1(\cdot), \ldots, \phi_L(\cdot)]$, we use $\boldsymbol{\phi}(i)$ to denote element-wise evaluations at i, i.e., $[\phi_1(i), \ldots, \phi_L(i)]$.

2.1 Multi-secret VSS

A multi-secret verifiable secret-sharing (VSS) protocol allows a dealer $\mathcal{D} \in [n]$ to share L secrets $\boldsymbol{z} = [z_1, \ldots, z_L]$ with all nodes. At the end of the protocol, each node $i \in [n]$ outputs the share $\boldsymbol{s}_i = [s_{i,1}, \ldots, s_{i,L}]$ along with a bit $b \in \{0, 1\}$. The bit b indicates whether the overall sharing was successful. In other words, $b = 1$ means that all honest nodes have successfully received their shares. The protocol must satisfy the following properties:

- *(Weak) guaranteed output.* If an honest node outputs $b = 1$, then every honest node i has a non-empty share $\boldsymbol{s}_i \neq \perp$.
- *Commitment.* There exist L polynomials $\phi_1(\cdot), \ldots, \phi_L(\cdot)$ all with degree f s.t. for any honest node i, if i has a non-empty share $\boldsymbol{s}_i \neq \perp$, then for all $k \in [L]$, $s_{i,k} = \phi_k(i)$.
- *Validity.* If the dealer is honest, then all honest nodes output $b = 1$, and the L polynomials (defined by the commitment property) satisfy $\phi_k(0) = z_k$ for all $k \in [L]$.
- *Secrecy (informal).* If the dealer is honest, the adversary learns no information about $\boldsymbol{z}$ beyond public values.

Note that it is possible that a node outputs nothing, which we capture with outputting $\boldsymbol{s}_i = \perp$. We formally define the secrecy property in §5.3, when we define the ideal functionality $\mathcal{F}_{\text{VSS}}$ for multi-secret VSS. Although $\mathcal{F}_{\text{VSS}}$ also captures the correctness properties defined above, we provide these property-based definitions for ease of exposition.

Remark on guaranteed output. The classical definition of VSS [6, 15] requires a stronger guaranteed output property, i.e., at the end of the protocol, either every honest node outputs the correct share, or nobody outputs (i.e., outputs $\perp$). To achieve this property, all existing VSS protocols use a broadcast [20]. However, the stronger guaranteed output property is not always required. For example, applications such as [8, 18] (including our DPSS protocol) do not need this strong guarantee. In this paper, we use a weaker guarantee on output, which lets us design a VSS protocol with $O(1)$ round complexity. We also note that a protocol satisfying our VSS definition can be easily extended to the classical definition by invoking a binary Byzantine agreement [1, 30, 33] at the end.

2.2 KZG commitment

We now describe the part of the KZG polynomial commitment required to understand our paper and refer readers to [27] for more details. To commit to a polynomial of degree d, the commitment scheme requires a powers-of-tau CRS $[g, g^{\tau}, \ldots, g^{\tau^d}]$ for a secret $\tau \in \mathbb{Z}_q^*$, and provides the following interfaces.

- $v \leftarrow \mathsf{Commit}(\phi(\cdot))$. On input a polynomial $\phi(\cdot)$ of degree d, it computes the commitment $v = g^{\phi(\tau)}$.
- $w_i \leftarrow \mathsf{CreateWitness}(\phi(\cdot), i)$. On input a polynomial $\phi(\cdot)$ and an index i, it computes a *witness* w_i for the evaluation $\phi(i)$ as:

$$w_i = g^{\psi(\tau)} \quad \text{where} \quad \psi(x) = \frac{\phi(x) - \phi(i)}{x - i}$$

- $b \leftarrow \mathsf{VerifyEval}(v, i, \phi(i), w_i)$. On input a polynomial commitment v, an index i, an evaluation $\phi(i)$, and the witness w_i, it checks whether $\phi(i)$ is equal to the committed polynomial evaluated at i.

Assuming the hardness of the Strong Diffie-Helman (SDH), the KZG commitment is *binding* and *hiding*, where.

- *(Evaluation) binding.* No adversary can compute two different polynomial evaluations $\phi(i)$ and $\phi'(i) \neq \phi(i)$ along with witnesses w_i and w_i' s.t. they are both verified by VerifyEval with respect to the same commitment v.
- *Hiding.* Given evaluations of a polynomial $\phi(\cdot)$ on any set X of less than d indices along with the witnesses and the commitment, no adversary can compute $\phi(j)$ for $j \notin X$.

Definition 2.1 (SDH Assumption). Let $\tau \in \mathbb{Z}_q^*$ be a random field element. For any $\ell \in O(\mathsf{poly}(\kappa))$ and any PPT adversary $\mathcal{A}$, the probability

$$\Pr[\mathcal{A}([g, g^{\tau}, \ldots, g^{\tau^{\ell}}]) \rightarrow (g^{\frac{1}{\tau+c}}, c)]$$

is negligible for any freely chosen $c \in \mathbb{Z}_q$.

2.3 Other primitives

We use a random oracle denoted $\mathsf{H}(\cdot)$ instantiated by a cryptographic hash function. We also use the random oracle to build a pseudorandom function (PRF). For simplicity, we use $\mathsf{PRF}_s(\mu)$ to denote $\mathsf{H}(s|\mu)$. Our protocol also uses the Feldman commitment [21] and a Gradecast protocol, as defined below.

Feldman commitment. For a polynomial $\phi(x) = a_0 + a_1 x + , \ldots, a_d x^d$ of degree-d, the Feldman commitment $\boldsymbol{v}$ is the vector defined as

$$\boldsymbol{v} = [g^{a_0}, \ldots, g^{a_d}]$$

Given the commitment $\boldsymbol{v}$, a evaluation $\phi(i)$ can be verified by checking that

$$g^{\phi(i)} = \prod_{0 \leq k \leq d} v_k^{i^k}$$

Gradecast [29]. Gradecast allows a dealer $\mathcal{D}$ to broadcast a message M with weak consensus. Specifically, at the end of the protocol, each node outputs (M, b) where $M \in \{0,1\}^L$ is a message of any length $L = O(\mathsf{poly}(\kappa))$ and $b \in \{0, 1\}$ is a *grade* bit satisfying the following properties.

- *Consistency.* If two honest nodes output $(M, *)$ and $(M', *)$, respectively, for $M, M' \neq \perp$ then $M = M'$.
- *Graded agreement.* If an honest node outputs $(M, 1)$, all honest nodes output $(M, *)$.
- *Validity.* If the dealer is honest, all honest nodes output $(M, 1)$.

Note that a gradecast allows a node to output nothing, which is expressed as outputting $M = \perp$. Concretely, we use the gradecast protocol (henceforth denoted GC) in Appendix A. GC has a communication cost $O(Ln + \kappa n^2)$.

3 OVERVIEW

In this section, we present an overview of this work to elaborate on the technical contributions.

3.1 Degree-binding KZG Commitment for Multiple Polynomials

As we describe in §1, the degree binding property intuitively guarantees that the committed polynomial is of the claimed degree (or lower), which is supposed to be the reconstruction threshold of the VSS scheme. Formalizing the degree binding property turns out to be non-trivial. Recall from §2.2, the KZG commitment of a polynomial $\phi(\cdot)$ of degree d is $v = g^{\phi(\tau)}$. However, v is also a commitment to a different polynomial $\phi'(\cdot)$ with degree $d' \neq d$, whenever $\phi(\tau) = \phi'(\tau)$. A tempting but incorrect way of defining degree binding is:

"No PPT adversary $\mathcal{A}$ can output a polynomial commitment v along with $d'+1$ evaluations and valid proofs such that interpolating the $d'+1$ evaluations results in a polynomial of degree $d' > d$."

The strong correctness property in the KZG paper [27, 28] is indeed defined in this flavor. However, this is insufficient for the VSS correctness due to its implicit constraint on d'. More concretely, as defined above, the PPT algorithm $\mathcal{A}$ needs to output $\Omega(d')$ values, which implicitly assumes d' is polynomial in the security parameter κ. In VSS, however, the adversary $\mathcal{A}$ (or the corrupt dealer) is required to compute only $O(n)$ evaluations (shares for nodes), even when it commits to a polynomial of super-polynomial degree, i.e., $d' = \omega(\mathsf{poly}(\kappa))$. We address this subtlety by defining the degree binding as follows.

Definition 3.1 (informal). No PPT adversary $\mathcal{A}$ can compute a commitment v to a polynomial $\phi(\cdot)$ together with

(1) A set X of $d+1$ indices and for each $i \in X$, the evaluation $\phi(i)$ and the corresponding witness; and
(2) An additional evaluation $\phi(j)$ for $j \notin X$ such that $\phi(j) \neq \phi'(j)$, where $\phi'(\cdot)$ is the degree-d polynomial defined by the evaluations of $\phi(\cdot)$ at X.

Intuitively, our definition ensures that even if v is a commitment to a polynomial of super-polynomial degree, all witness evaluations of $\phi(\cdot)$ lie on a unique polynomial $\phi'(\cdot)$ of degree d.

We note that the proof technique used for strong correctness in the KZG paper [28] does not apply to Definition 3.1 (see Appendix B for more detailed discussions). We also note that our degree binding property is equivalent to the interpolation binding defined in [2], but with a slightly different description for ease of extension to the batch setting below.

Degree-binding for batch setting. As noted in §1, within the context of VSS, KZG commitment is specifically useful in batch

settings where a linear number of polynomials need to be committed. Therefore, we will define the concept of degree binding for multiple polynomials. To this end, we first extend the interface of the KZG commitment to include a *degree proof*. Specifically, for a set of polynomials $\boldsymbol{\phi} = [\phi_1(\cdot), \ldots, \phi_L(\cdot)]$, a degree proof π convinces a verifier that all polynomials in $\boldsymbol{\phi}$ are of degree at most d. We extend the degree binding property for multiple polynomials as follows.

Definition 3.2 (informal). No adversary $\mathcal{A}$ can compute a vector of commitments $\boldsymbol{v}$ to a vector of polynomials $\boldsymbol{\phi}$ together with

(1) A set of X of $d+1$ indices, and for each $i \in X$, the evaluations $\boldsymbol{\phi}(i)$ with the witnesses, and the degree proof π; and
(2) An additional single evaluation $\phi_k(j) \neq \phi'_k(j)$ with the witness for any $k \in [L]$ where $\phi'_k(\cdot)$ is a degree-d polynomial interpolated from the $d+1$ evaluations for $\phi_k(\cdot)$ at X.

Intuitively, in VSS with $d = f$, the above property guarantees that when $f+1$ honest shareholders receive all L shares with a valid degree proof, the shares of the remaining honest nodes lie on the same degree-f polynomial. Looking ahead, this is sufficient to achieve VSS correctness.

Degree proof from aggregated commitment. We obtain a degree proof from an aggregation of linear-sized commitments. A simple example is an aggregated Feldman commitment. Specifically, the degree proof π for polynomials $\phi_1(\cdot), \ldots, \phi_L(\cdot)$ of the same degree is a Feldman commitment to the following aggregated polynomial

$$\Phi(\cdot) = \sum_{1 \leq k \leq L} \rho_k \cdot \phi_k(\cdot),$$

which is a random linear combination of the polynomials $\boldsymbol{\phi}$ with coefficients $\rho_1, \ldots, \rho_L$; let us assume here these random values are chosen after the polynomials are given (we apply Fiat-Shamir later [22]). In VSS, a dealer, when sharing L secrets over polynomials $\phi_1(\cdot), \ldots, \phi_L(\cdot)$, also sends the above Feldman commitment along with the shares. Once a shareholder i receives the shares $\phi_1(i), \ldots, \phi_L(i)$, it verifies the aggregation of the shares $\Phi(i)$ with the Feldman commitment, besides the verification of individual share with the KZG commitment. The Feldman commitment has $\Omega(\kappa n)$ size (as the degree is $d = f$), but it is amortized over $L = O(n)$ secrets. Due to the randomization, it is hard for a corrupt dealer to choose dependent polynomials with higher degrees (trying to cancel out the higher degree terms) while preserving the aggregated polynomial degree-f. While the construction is quite intuitive, the proof is not as straightforward as one might expect. Our proof based on a reduction to the SDH problem (in Lemma 4.2) might be of independent interest.

The key conceptual difference from the previous approach to achieving the degree binding of KZG lies in not explicitly preventing an adversary from computing a KZG commitment to a high-degree polynomial. Instead, we separately detect the degree by making use of a linear-sized commitment that is degree-revealing by design. This approach, not only helps us eliminate idealized group assumptions but also enables us to reuse the same powers-of-tau CRS for multiple degrees. Specifically, we can support polynomials with any degree $0 < d \leq m$ using a single CRS $[g, g^{\tau}, \ldots, g^{\tau^m}]$, eliminating the need for repeated setup.

Why not use aggregated Feldman directly in place of KZG? One might wonder why we do not use the aggregated Feldman commitment directly to verify the shares in place of KZG commitments. This is because the verification using the aggregated commitment requires having shares of all the committed polynomials. In other words, it does not allow for the individual verification of each share. This limitation is problematic because, in most applications, there is a need to utilize each individual secret share separately. Therefore, we opt to use the aggregated commitment only for degree checking while using KZG commitments for verifying each share.

3.2 Multi-Secret VSS with Optimal Communication

One of the major tasks in VSS is to disseminate the shares efficiently but verifiably. Namely, we have to solve the following problem with $O(L)$ communication (i.e., constant cost per node): an honest node i receives the correct share $\boldsymbol{s}_i = [\phi_1(i), \ldots, \phi_L(i)]$ or all honest nodes detect the corrupt dealer. Note that we will have n instances of this task (i.e., for each node i) so the overall cost will be $O(Ln)$. Below, let us assume for simplicity that the KZG commitments to the polynomials are known to all nodes. Also, we assume $L = f + 1$ for simplicity.

The first natural technique we can use is a verifiable information dispersal algorithm (IDA) [12, 36]. In IDA, there is a single sender who holds a message M, and at the end of the protocol, the erasure-coded symbols of the message are distributed among the nodes. Specifically, the message M is encoded into n symbols $[c_1, \ldots, c_n]$ using (f, n)-erasure coding, and each node j will receive j-th symbol c_j. The dealer in VSS can use IDA to disseminate the node i's share $M = \boldsymbol{s}_i$. Upon receiving the assigned symbol c_j, each node j forwards it to the shareholder i and sends a vote to all nodes. If $2f+1$ nodes send votes, at least $f+1$ honest nodes must have forwarded their code words, so the shareholder i can successfully reconstruct the original message. Otherwise, all honest nodes know the dealer is corrupt. Since each symbol is of constant size, the communication cost is $O(L)$ (the voting cost is amortized over n instances of dissemination).

However, there is one remaining task to complete the problem. Recall that the message M that node i receives must be the correct batch of shares $\boldsymbol{s}_i = [\phi_1(i), \ldots, \phi_L(i)]$. This requirement is not guaranteed in the IDA above. In other words, while node i is guaranteed to receive a message M that the dealer has sent, the message M may not necessarily contain the correct shares – potentially, an arbitrary blob when the dealer is corrupt. Therefore, if the dealer sends an invalid message M through the IDA, the corresponding shareholder i must forward the message M to all other nodes to help other honest nodes detect the corrupt dealer. However, this dealer implication step incurs $\Omega(\kappa n^3)$ communication in the worst case (i.e., when all nodes implicate the dealer) as the message M is of size $L = \Omega(n)$ [41].

Efficient implication from systematic RS code. To achieve the dealer implication step with $O(\kappa n^2)$ communication, let us delve more into the IDA implementation. The problem with the use of black-box IDA is that it is hard to validate the message without reconstructing the whole message. Each node can send only a constant-sized message during the implication step. In the above

IDA using black-box erasure coding, each individual code symbol does not carry sufficient information to detect an invalid message. We solve this problem by utilizing a specific erasure coding scheme in which each code symbol has a certain relation to the original message. Concretely, we implement IDA using a *systematic* Reed-Solomon code [37]. The message $M = \boldsymbol{s}_i$ is encoded as evaluations of a degree-f polynomial $\psi(\cdot)$ interpolated from the shares

$$\boldsymbol{s}_i = [\phi_1(i), \ldots, \phi_L(i)].$$

Namely, $\psi(k) = \phi_k(i)$ for all $k \in [L]$. Then the code words are defined as

$$\boldsymbol{c} = [\psi(1), \ldots, \psi(n)].$$

Note that the first L symbols correspond to the original message (by the nature of systematic code).

$$c_1 = \phi_1(i), c_2 = \phi_2(i), \ldots, c_L = \phi_L(i).$$

Recall that the message M is considered invalid if any single element $s_{i,k}$ is not a valid share $\phi_k(i)$. Therefore, receiving the corresponding symbol $c_k = s_{i,k}$ is enough to detect an invalid message M, and hence a corrupt dealer. This allows the shareholder i to implicate the dealer with a constant-sized message and reduces the overall communication cost to $O(\kappa n^2)$.

Note that the protocol described above does not guarantee the secrecy of the shares. Our protocol in §5 applies a simple one-time pad based on PRF to hide the shares during IDA.

4 DEGREE BINDING KZG COMMITMENT

In this section, we augment the KZG commitment to make it degree-binding. The augmented KZG commitment supports polynomials of any degree $0 < d \leq m$ under a universal structured reference string $[g, g^{\tau}, \ldots, g^{\tau^m}]$ for a trapdoor $\tau \in \mathbb{Z}_q^*$.

Extended interface. To formally define degree binding and our commitment scheme, we first extend the interface of the KZG commitment. Specifically, in addition to all the interfaces of the original KZG commitment, we define two auxiliary functions to prove/verify the degree of the committed polynomials as follows:

- $\pi \leftarrow \mathsf{ProveDeg}(\boldsymbol{\phi}, \boldsymbol{v})$. It takes as input L polynomials $\boldsymbol{\phi}$ of degree at most d, and the corresponding commitments $\boldsymbol{v}$,
$$\boldsymbol{\phi} = [\phi_1(\cdot), \ldots, \phi_L(\cdot)]$$
$$\boldsymbol{v} = [g^{\phi_1(\tau)}, \ldots, g^{\phi_L(\tau)}]$$
and outputs a degree proof π.
- $b \leftarrow \mathsf{VerifyDeg}(d, \boldsymbol{v}, \pi, i, \boldsymbol{\phi}(i))$. It takes as input the degree d, a vector $\boldsymbol{v}$ of commitments, the degree proof π, and the evaluations of all committed polynomials on index i, and outputs $b \in \{0, 1\}$ indicating if all the polynomials are of degree at most d.

Degree binding. Our goal is to design an extended KZG commitment with the above auxiliary functions that satisfies the following degree binding property:

Definition 4.1 (Degree Binding). For any $d > 0$ and any PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ successfully computes all of the following simultaneously is negligible.

(1) Commitments $\boldsymbol{v} = [v_1, \ldots, v_L]$ and a degree proof π.

(2) A set H of $d+1$ distinct indices, evaluations $\boldsymbol{\phi}(i)$ along with the witnesses $w_{i,1}, \ldots, w_{i,L}$ for all $i \in H$ s.t.
$$\mathsf{VerifyEval}(v_k, i, \phi_k(i), w_{i,k}) = 1 \ \ \forall k \in [L]$$
$$\mathsf{VerifyDeg}(d, \boldsymbol{v}, \pi, i, \boldsymbol{\phi}(i)) = 1$$

(3) For any $k \in [L]$ and $j \in \mathbb{Z}_q$, a polynomial evaluation $\phi_k(j)$ along with the witness $w_{j,k}$ s.t.
$$\mathsf{VerifyEval}(v_k, i, \phi_k(j), w_{j,k}) = 1$$
$$\phi_k(j) \neq \phi'_k(j)$$
where $\phi'_k(\cdot)$ is the degree-d polynomial uniquely defined by the $d+1$ evaluations of $\phi_k(\cdot)$ on indices in H.

Intuition in the use of VSS. In our VSS protocol, a dealer shares L secrets $z_1, \ldots, z_L$ over polynomials $\phi_1(\cdot), \ldots, \phi_L(\cdot)$ all with degree at most $d = f$. A corrupt dealer may try to use polynomials with a higher degree $d > f$. However, our protocol makes sure at least $f+1$ honest nodes verify their shares with VerifyEval and VerifyDeg. These shares define H and $\phi'_k(\cdot)$ for all $k \in [L]$. The degree binding property guarantees, if another node j receives its share $\phi_k(\cdot)$ of any k-th secret, then it must be $\phi_k(j) = \phi'_k(j)$. Therefore, all valid shares lie on a unique polynomial of degree at most f.

Our augmented KZG commitment. We now describe our augmented KZG commitment with the two auxiliary functions in Figure 1. The proving function ProveDeg generates the Feldman commitment to the aggregated polynomial. Specifically, it first generates deterministic pseudo-random values $\rho_1, \ldots, \rho_L$ that bind to the commitment $\boldsymbol{v}$ by querying the random oracle, i.e., $\rho_k = \mathsf{H}(\boldsymbol{v}|k)$. Then, it computes a commitment π to the aggregated polynomial

$$\Phi(\cdot) = \sum_{1 \leq k \leq L} \rho_k \cdot \phi_k(\cdot) = \alpha_0 + \alpha_1 x + \ldots + \alpha_d x^d.$$

The verification function VerifyDeg performs the same aggregation operation on the shares and the KZG commitments. Specifically, it first performs the aggregation on the shares in the exponent to check that

$$\sum_{1 \leq k \leq L} \rho_k \cdot \phi_k(i) = \alpha_0 + \alpha_1 i + \ldots + \alpha_d i^d,$$

It then performs the aggregation on the commitments on the pairing to verify

$$\sum_{1 \leq k \leq L} \rho_k \cdot \phi_k(\tau) = \alpha_0 + \alpha_1 \tau + \ldots + \alpha_d \tau^d.$$

We show that our augmented KZG commitment is degree-binding under the SDH assumption.

Lemma 4.2. *Under the SDH assumption, the augmented KZG commitment in Figure 1 is degree binding.*

Proof. Suppose an adversary $\mathcal{A}$ computes (1) and (2) in Definition 4.1. Namely, $\mathcal{A}$ has commitments $\boldsymbol{v} = [v_1, \ldots, v_L]$, a degree proof π, and evaluations $\boldsymbol{\phi}(i)$ with the witnesses $w_{i,1}, \ldots, w_{i,L}$ for $d+1$ distinct indices $i \in H$. Let $\beta(\cdot)$ be the degree-d polynomial defined by the proof $\pi = [g^{\alpha_0}, \ldots, g^{\alpha_d}]$:

$$\beta(x) = \alpha_0 + \ldots + \alpha_d x^d$$

Degree-binding KZG commitment.

Besides all functionalities of KZG commitment, we have two additional functionalities to prove and verify the degree d of polynomials.

$\pi \leftarrow \mathsf{ProveDeg}(\boldsymbol{\phi}, \boldsymbol{v})$.

- Generate random values $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_L]$ by querying the random oracle $\rho_k \leftarrow \mathsf{H}(\boldsymbol{v}|k)$ for each $k \in [L]$.
- Let $\boldsymbol{a}_k$ be the vector of coefficients of $\phi_k(\cdot)$. Compute
$$[\alpha_0, \ldots, \alpha_d] = \rho_1 \boldsymbol{a}_1 + \ldots + \rho_L \boldsymbol{a}_L.$$
- Output $\pi = [g^{\alpha_1}, \ldots, g^{\alpha_d}]$.

$b \leftarrow \mathsf{VerifyDeg}(d, \boldsymbol{v}, \pi, i, \boldsymbol{\phi}(i))$.

- If $|\pi| \neq d$, then output $b = 0$.
- Generate random values $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_L]$ by querying the random oracle $\rho_k \leftarrow \mathsf{H}(\boldsymbol{v}|k)$ for each $k \in [L]$.
- If both of the following conditions hold, then output $b = 1$, otherwise, $b = 0$.
$$\prod_{1 \le k \le L} g^{\rho_k \cdot \phi_k(i)} = \prod_{0 \le j \le d} \pi_j^{i^j}$$
$$\prod_{1 \le k \le L} e(v_k, g)^{\rho_k} = \prod_{0 \le j \le d} e(\pi_j, g^{\tau^j})$$

π_j is the j-th element of π, and $e(\cdot)$ is the pairing.

Figure 1: Our augmented KZG commitment.

Let $\Phi'(\cdot)$ be the polynomial defined as follows:

$$\Phi'(\cdot) = \sum_{1 \le k \le L} \rho_k \cdot \phi'_k(\cdot).$$

where $\phi'_k(\cdot)$ is the degree-d polynomial interpolated from the $d+1$ evaluations on $\phi_k(\cdot)$ for H. Due to the first condition of VerifyDeg, and since $\phi_k(i) = \phi'_k(i)$ for each $i \in H$ by definition, we have

$$\prod_{1 \le k \le L} g^{\rho_k \cdot \phi'_k(i)} = \prod_{0 \le j \le d} g^{\alpha_j \cdot i^j}.$$

Thus, we have $\Phi'(i) = \beta(i)$ for all $i \in H$. Since both $\Phi'(\cdot)$ and $\beta(\cdot)$ are of degree-d and share the same points on $d+1$ indices, we have $\Phi'(\cdot) = \beta(\cdot)$. Due to the second condition of VerifyDeg, we have

$$\sum_{1 \le k \le L} \rho_k \cdot \phi_k(\tau) = \beta(\tau) = \Phi'(\tau).$$

Put in another way, let $\boldsymbol{\delta} = [\delta_1, \ldots, \delta_L]$ where $\delta_k = \phi_k(\tau) - \phi'_k(\tau)$. We have that the inner product of $\boldsymbol{\delta}$ and $\boldsymbol{\rho}$ is $\boldsymbol{\delta} \cdot \boldsymbol{\rho} = 0$. Since the choice of the vector $\boldsymbol{\delta}$ (uniquely determined by $\boldsymbol{v}$) is independent of the random oracle's outputs $\boldsymbol{\rho}$, the probability $\mathcal{A}$ can choose $\boldsymbol{\delta} \neq \mathbf{0}$ that satisfies $\boldsymbol{\delta} \cdot \boldsymbol{\rho} = 0$ is negligible. Therefore, $\boldsymbol{\delta} = \mathbf{0}$ and we have $\phi_k(\tau) = \phi'_k(\tau)$ for all $k \in [L]$.

Now, suppose for contradiction that $\mathcal{A}$ can also compute (3), namely, an evaluation with a witness

$$\phi_k(j), \;\; w_{j,k} = g^{\frac{\phi_k(\tau) - \phi_k(j)}{\tau - j}}$$

for some $k, j \in [L]$ that satisfies $\phi_k(j) \neq \phi'_k(j)$. The adversary $\mathcal{A}$ can also easily compute

$$\phi'_k(j), \;\; w'_{j,k} = g^{\frac{\phi'_k(\tau) - \phi'_k(j)}{\tau - j}}$$

Since we have $\phi_k(\tau) = \phi'_k(\tau)$, the adversary $\mathcal{A}$ can compute

$$\left(\frac{w_{j,k}}{w'_{j,k}}\right)^{\frac{1}{\phi'_k(j) - \phi_k(j)}} = g^{\frac{1}{\tau - j}}.$$

This breaks the SDH assumption. □

5 MULTI-SECRET VSS WITH OPTIMAL COMMUNICATION

This section presents a multi-secret VSS with $O(\kappa L n + \kappa n^2)$ communication for sharing L secrets. For simplicity, we first present a synchronous protocol, and then explain how to extend it to an asynchronous protocol using existing techniques.

Share format. Our VSS protocol (in fact VSS protocols in general) outputs not only the share $s_i = \phi(i)$ but the witness w_i and the commitment v to the polynomial. Therefore, for ease of presentation, we say (s_i, w_i, v) is *a valid share for* $i \in [n]$ if

$$\mathsf{VerifyEval}(v, i, s_i, w_i) = 1$$

When the shared polynomial $\phi(\cdot)$ is known, we say (s_i, w_i, v) is a valid share for i *over* $\phi(\cdot)$.

Encryption with PRF. We make use of PRF to generate a one-time pad for encrypting each node's share. We assume each node i before starting the protocol exchanges a secret key $\mathsf{sk}_i \in \mathbb{Z}_q^*$ with the dealer $\mathcal{D}$ and receives the dealer's signature on the secret key. For simplicity, we use the notation $\rho_{i,k} = \mathsf{PRF}_{\mathsf{sk}_i}(k)$ and $\rho'_{i,k} = \mathsf{PRF}_{\mathsf{sk}_i}(n+k)$ to denote the one-time pads for $k \in [n]$.

5.1 Our Protocol

We describe our multi-secret VSS protocol (denoted VSS) in Figure 2. For ease of exposition, we present a protocol for sharing $L = f + 1$ secrets, and it can be easily extended to support any number of secrets.

Intuitive overview. The protocol consists of three phases. The dealer $\mathcal{D}$ sends to each node i the share $\boldsymbol{\phi}(i)$ through a verifiable information dispersal algorithm (IDA) (Commit–Reconstruct). If a node receives an invalid message from the IDA, the node implicates the dealer (Accuse). Then, nodes vote in two rounds if they have not detected any misbehavior of the dealer and output the shares if there are enough votes (Ready–Output). These steps basically follow hbACSS [41]. There are, however, two key differences from hbACSS as we alluded to in §3. First, we use our augmented KZG commitment (from §4), which helps achieve the *commitment* property of VSS. Second, our IDA implementation is based on the systematic RS code, which allows nodes to implicate a corrupt dealer with a constant-sized accusation message. We elaborate on each step below.

Commit. The dealer $\mathcal{D}$ first computes the sharing polynomial, encoding polynomials (for erasure coding), and the associated commitments and degree proofs. Specifically, it first samples the random sharing polynomials of degree-f denoted $\phi_1(\cdot), \ldots, \phi_{f+1}(\cdot)$

VSS – Multi-secret VSS.

Let $\mathcal{D}$ be the dealer who has secrets $\boldsymbol{z} := [z_1, \ldots, z_{f+1}]$ to share.

- We assume each node i and the dealer $\mathcal{D}$ share a random secret key sk_i signed by $\mathcal{D}$.
- Let $\rho_{i,k} = \mathsf{PRF}_{\mathsf{sk}_i}(k)$ and $\rho'_{i,k} = \mathsf{PRF}_{\mathsf{sk}_i}(n+k)$ denote the one-time pads for each $k \in [n]$.

// Round 1–3.

Commit. The dealer $\mathcal{D}$ computes the following:

(1) $f+1$ random polynomials $\phi_1(\cdot), \ldots, \phi_{f+1}(\cdot)$ with degree f for sharing $z_1, \ldots, z_{f+1}$.
(2) For each $j \in [n]$, let $\psi_j(\cdot)$ and $\psi'_j(\cdot)$ be two polynomials with degree-f s.t. for all $k \in [f+1]$

$$\psi_j(k) = \phi_k(j) \oplus \rho_{j,k} \quad \psi'_j(k) = w_{j,k} \oplus \rho'_{j,k}$$

where $w_{j,k}$ is the witness for $\phi_k(j)$.
(3) Let $\boldsymbol{v}, \boldsymbol{u}, \boldsymbol{u'}$ be the vectors of commitments to $\boldsymbol{\phi}(\cdot), \boldsymbol{\psi}(\cdot), \boldsymbol{\psi'}(\cdot)$, and π_v, π_u, π'_u be the associated degree-proofs.

The dealer $\mathcal{D}$ sends to each node $j \in [n]$, for all $k \in [n]$,

$$\mathsf{code}_{k,j} := (\psi_k(j),\ \psi'_k(j),\ \mu_{j,k},\ \mu'_{j,k})$$

where $\mu_{j,k}$ and $\mu'_{j,k}$ are the witnesses for $\psi_k(j)$ and $\psi'_k(j)$.
$\mathcal{D}$ also sends $(\boldsymbol{v}, \boldsymbol{u}, \boldsymbol{u'}, \pi_v, \pi_u, \pi'_u)$ to all nodes through GC.

// Round 4.

Forward. If node i has received $\mathsf{code}_{j,i}$ for all $j \in [n]$ that are verified with the commitments $\boldsymbol{u}, \boldsymbol{u'}$ and degree proofs π_u, π'_u received from GC with grade $b = 1$, then node i forwards $\mathsf{code}_{j,i}$ to node j.

// Round 5.

Reconstruct. Node i computes $\psi_i(\cdot)$ and $\psi_i(\cdot)$ by interpolation using $f+1$ points received through $\mathsf{code}_{i,*}$ verified with the commitments $\boldsymbol{u}, \boldsymbol{u'}$ and the proofs π_u, π'_u received from GC (with any grade).

Accuse. Node i verifies that for all $k \in [L]$, $(s_{i,k}, w_{i,k}, v_k)$ is a valid share for i, where

$$s_{i,k} = \psi_i(k) \oplus \rho_{i,k} \quad w_{i,k} = \psi'_i(k) \oplus \rho'_{i,k}$$

If the verification failed for any k, node i sens to all nodes

$$\mathsf{accuse}_i := (\psi_i(k), \psi'_i(k), \mu_{i,k}, \mu'_{i,k}),$$

for an arbitrary such $k \in [n]$ along with the signed sk_i.

// Round 6.

Ready. Node i sends "ready" to all nodes if

(1) i forwarded $\mathsf{code}_{j,i}$ to each $j \in [n]$ in round $t = 3$; and
(2) Node i's shares $(\boldsymbol{s}_i, \boldsymbol{w}_i, \boldsymbol{v})$ are verified with π_v; and
(3) i has not received any valid accusation, namely

$$\mathsf{accuse}_j := (\hat{s}_{j,k}, \hat{w}_{j,k}, \mu_{j,k}, \mu'_{j,k}).$$

with the signed sk_j s.t. both of the following hold.
(a) $(\hat{s}_{j,k}, \mu_{j,k}, u_k)$ and $(\hat{w}_{j,k}, \mu'_{j,k}, u'_k)$ are both valid shares for index k.
(b) $(s_{j,k}, w_{j,k}, v_j)$ is not a valid share for j where

$$s_{j,k} = \hat{s}_{j,k} \oplus \rho_{j,k} \quad w_{j,k} = \hat{w}_{j,k} \oplus \rho'_{j,k}$$

// Round 7.

Complete. If node i has received "ready" from $2f+1$ nodes, then send "complete" to all nodes.

// At the end of round 7.

Output. If node i has received "complete" from $m > 2f$ nodes, then i outputs $b = 1$, otherwise $b = 0$. If $m > f$, then i outputs the share $(\boldsymbol{s}_i, \boldsymbol{w}_i, \boldsymbol{v})$.

Figure 2: Our multi-secret VSS for sharing $L = f+1$ secrets. For simplicity of presentation, we assume the representation of a group element has the same length as that of a field element.

s.t. $\phi_k(0) = z_k$ for all $k \in [f+1]$. Then, for each node $i \in [n]$, the dealer computes two encoding polynomials $\psi_i(\cdot)$ and $\psi'_i(\cdot)$ both with degree-f for disseminating the shares and the associated witnesses. They are interpolated from the $f+1$ shares/witnesses encrypted with one-time pads. For each $k \in [f+1]$,

$$\psi_i(k) = \phi_k(i) \oplus \rho_{i,k} \quad \psi'_i(k) = w_{i,k} \oplus \rho'_{i,k},$$

where $w_{i,k}$ is the witness for $\phi_k(i)$. Here we note that each witness is a group element and its representation is larger than a field element (e.g., a share), so we have to use multiple encoding polynomials depending on the representation of a group element. For simplicity, we assume a witness has the same length as a field element.

The dealer also computes the vectors $\boldsymbol{v}, \boldsymbol{u}, \boldsymbol{u'}$ of commitments to the three types of polynomials. Specifically, for each $k \in [f+1]$, v_k is the commitment to $\phi_k(\cdot)$, and for each $k \in [n]$, u_k and u'_k are the commitments to $\psi_k(\cdot)$ and $\psi'_k(\cdot)$, respectively. Also π_v, π_u, and π'_u are degree proofs for $\boldsymbol{\phi}(\cdot)$, $\boldsymbol{\psi}(\cdot)$, and $\boldsymbol{\psi'}(\cdot)$, respectively. Specifically,

$$\pi_v = \mathsf{ProveDeg}([\phi_1(\cdot), \ldots, \phi_{f+1}(\cdot)], \boldsymbol{v}).$$

π_u and π'_u are computed similarly. Then, the dealer starts disseminating the share for node i by sending to each node j the code word (for both share and witness)

$$\mathsf{code}_{i,j} := (\psi_i(j),\ \psi'_i(j),\ \mu_{j,i},\ \mu'_{j,i})$$

where $\mu_{j,i}$ and $\mu'_{j,i}$ are the witnesses for $\psi_i(j)$ and $\psi'_i(j)$. The dealer also sends to all nodes the commitments and the degree proofs by invoking a gradecast (defined in §2), denoted GC.

Forward and reconstruct. If the gradecast GC outputs the commitments and proofs with grade $b = 1$, a node verifies the assigned code words. Specifically, for the encoding polynomials $\boldsymbol{\psi}(\cdot) = [\psi_1(\cdot), \ldots, \psi_n(\cdot)]$, node i checks that

$$\mathsf{VerifyEval}(u_j, i, \psi_j(i), \mu_{j,i}) = 1 \ \ \forall j \in [n]$$
$$\mathsf{VerifyDeg}(f, \boldsymbol{u}, \pi_u, i, \boldsymbol{\psi}(i)) = 1,$$

Node i also performs the same check on $\boldsymbol{\psi}'(i)$. If all of the verifications pass, node i forwards to node each j the code word $\mathsf{code}_{j,i}$.

Node i then computes the encoding polynomials $\psi_i(\cdot)$ and $\psi'_i(\cdot)$ by interpolation using the collected code words after verifying with the commitments u_i and u'_i. This completes the IDA for node i.

Accuse. The node i then checks if the reconstructed message is valid. Specifically, for each $k \in [f+1]$, node i performs decryption on the encoded symbols

$$s_{i,k} = \psi_i(k) \oplus \rho_{i,k} \quad w_{i,k} = \psi'_i(k) \oplus \rho'_{i,k}$$

and then check if $(s_{i,k}, w_{i,k}, v_k)$ is a valid share. If any of them are invalid, node i chooses an arbitrary such k, and implicates the dealer by sending to all nodes

$$\mathsf{accuse}_i := (\psi_i(k), \psi'_i(k), \mu_{i,k}, \mu'_{i,k}),$$

and reveals the PRF key sk_i signed by the dealer.

Ready/complete/output. The sharing is completed after two rounds of voting. Each node first sends "ready" if it has not detected any dealer's misbehavior. The absence of misbehavior is checked based on three criteria. First, the node must have forwarded all code words supposed to be assigned. Second, the share $(\boldsymbol{s}_i, \boldsymbol{w}_i, \boldsymbol{v})$ must be verified with the degree-proof π_v, namely

$$\mathsf{VerifyDeg}(f, \boldsymbol{v}, \pi_v, i, \boldsymbol{s}_i) = 1,$$

Finally, the node i must have neither sent nor received any valid accusation. If all of these conditions hold, then the node considers the dealer has behaved honestly. Each node then sends "complete" if it receives "ready" from $2f+1$ nodes. Finally, node i outputs the share $(\boldsymbol{s}_i, \boldsymbol{w}_i, \boldsymbol{v})$ if it has received "complete" from more than f nodes. Also, if $2f+1$ nodes sent "complete", then outputs $b = 1$ indicating the overall sharing is successful.

5.2 Correctness Proof

We first show the correctness of our multi-secret VSS protocol for sharing $L = f+1$. Validity is straightforward.

Lemma 5.1 (Commitment). *There exist polynomials $\phi_1(\cdot), \ldots, \phi_L(\cdot)$ all with degree f s.t. for any honest node $i \in [n]$, if i has a non-empty output $(\boldsymbol{s}_i, \boldsymbol{w}_i, \boldsymbol{v}) \neq \perp$ then, for all $k \in [L]$, $(s_{i,k}, w_{i,k}, v_k)$ is a valid share for i over $\phi_k(\cdot)$.*

Proof. Suppose an honest node i outputs $(\boldsymbol{s}_i, \boldsymbol{w}_i, \boldsymbol{v}) \neq \perp$. Then, for each $k \in [L]$, $(s_{i,k}, w_{i,k}, v_k)$ is a valid share for node i. The node i must have received the commitments $\boldsymbol{v}$ from GC. Due to the consistency property of GC, honest nodes do not have any other commitments $\boldsymbol{v}' \neq \boldsymbol{v}$. Therefore, if any honest node j has an output $(\boldsymbol{s}_j, \boldsymbol{w}_j, \boldsymbol{v}') \neq \perp$, then, $\boldsymbol{v} = \boldsymbol{v}'$ and for each $k \in [L]$, $(s_{j,k}, w_{j,k}, v_k)$ is a valid share for node j. The honest node i computes the output $(\boldsymbol{s}_i, \boldsymbol{w}_i, \boldsymbol{v}) \neq \perp$ after receiving "complete" from at least $f+1$ nodes. Out of these $f+1$ nodes, at least one node is honest, who has received "ready" from at least $2f+1$ nodes. A subset H of $f+1$ nodes must be honest. Each node $j \in H$ must have received a unique valid share $(s_{j,k}, w_{j,k}, v_k)$ for each $k \in [L]$ (due to the evaluation binding property). The shares for H define, for each $k \in [L]$, a unique degree-f polynomial $\phi_k(\cdot)$. Since all of the shares for H are also verified by VerifyDeg, due to the degree-binding property, node i's output $(s_{i,k}, w_{i,k}, v_k)$ must be a share over $\phi_k(\cdot)$ for each $k \in [L]$. □

Lemma 5.2 (Guaranteed output). *If an honest node outputs $b = 1$, then all honest nodes have non-empty ($\neq \perp$) outputs.*

Proof. If an honest node outputs $b = 1$, at least $2f+1$ nodes must have sent "complete", out of which at least $f+1$ nodes (say H) must be honest. Let i be any honest node. Each honest node $j \in H$ must have forwarded to i

$$\mathsf{code}_{i,j} := (\psi_i(j), \psi'_i(j), \mu_{j,i}, \mu'_{j,i}),$$

and all of them must be verified with the commitments $\boldsymbol{u}, \boldsymbol{u}'$ and the degree proofs π_u, π'_u received from GC with grade $b = 1$. Due to the graded consistency property of GC, node i must receive the commitments $\boldsymbol{u}, \boldsymbol{u}'$ and the degree proofs π_u, π'_u from GC. Thus, the node i can interpolate the degree-f polynomials $\psi_i(\cdot)$ and $\psi'_i(\cdot)$ from the $f+1$ verified points on each polynomial.

Suppose any of the reconstructed share $(s_{i,k}, w_{i,k}, v_k)$ (where $s_{i,k} = \psi_k(i) \oplus \rho_{i,k}$ and $w_{i,k} = \psi'_k(i) \oplus \rho'_{i,k}$) is not a valid share for i, then node i would have sent to all nodes

$$\mathsf{accuse}_i := (s_{i,k}, w_{i,k}, \mu_{i,k}, \mu'_{i,k}).$$

Then, nodes in H would not have sent "ready", a contradiction. Therefore, the node i must have received, for all $k \in [L]$, a valid share $(s_{i,k}, w_{i,k}, v_k)$, and outputs $(\boldsymbol{s}_i, \boldsymbol{w}_i, \boldsymbol{v}) \neq \perp$. □

5.3 Secrecy Proof

Next, we show the secrecy of our protocol VSS based on the standard simulation-based argument [13, 14]. The ideal functionality is defined in Figure 3. We first briefly mention that the functionality satisfies the VSS correctness. If the dealer is honest, all honest nodes receive the shares of the secrets $\boldsymbol{z}$ over randomly sampled degree-f polynomials with success bit $b = 1$ (validity). Even if the dealer is corrupt, the functionality computes honest nodes' shares over a unique degree-f polynomial $\phi_k(\cdot)$ for each $k \in [L]$ (commitment). Furthermore, if there is an honest node that outputs success bit $b = 1$, then the functionality must receive $b^*_i \neq \perp$ for all i, so it delivers shares to all honest nodes (guaranteed output). We now show the secrecy of our protocol below.

Lemma 5.3. *The protocol* VSS *realizes the functionality $\mathcal{F}_{\mathsf{VSS}}$.*

Proof. Let $\mathcal{A}$ be the adversary, and $\mathcal{Z}$ be the environment. We construct a simulator $\mathcal{S}$ that simulates the real-world adversary's view in the execution of VSS while interacting with the functionality $\mathcal{F}_{\mathsf{VSS}}$. Without loss of generality, we assume nodes $[1, \ldots, f]$ are corrupt.

Corrupt dealer case. $\mathcal{S}$ locally executes VSS with $\mathcal{A}$. Let $\boldsymbol{\phi} = [\phi_1(\cdot), \ldots, \phi_L]$ be the degree-f polynomials used to share the secrets in the local execution. Due to the commitment property, a unique polynomial $\phi_k(\cdot)$ always exists for each $k \in [L]$. The simulator $\mathcal{S}$ computes the $\phi_k(\cdot)$ by interpolating shares for $f+1$ honest nodes who sent "ready". Let $\boldsymbol{b}^* = [b^*_1, \ldots, b^*_n]$ where $b^*_1, \ldots, b^*_f = 0$ and $b^*_{f+1}, \ldots, b^*_n$ be the success bit b that VSS outputs in honest nodes $f+1, \ldots, n$. $\mathcal{S}$ sends to $\mathcal{F}_{\mathsf{VSS}}$ both $\boldsymbol{\phi}$ and $\boldsymbol{b}$. $\mathcal{S}$ sends any message to the environment $\mathcal{Z}$ that $\mathcal{A}$ sends in the local execution.

Now we show $\mathcal{Z}$'s view in the ideal world is indistinguishable from that of the real world. First, honest nodes' outputs should be identical to the outputs in the simulated execution due to the

$\mathcal{F}_{\mathsf{VSS}}$ – Ideal functionality of VSS.

Let $L = f + 1$.

- In round 1, receive from an honest dealer $\boldsymbol{z} = [z_1, \ldots, z_L]$, and compute the following
  - Randomly sampled polynomials $\boldsymbol{\phi} = [\phi_1(\cdot), \ldots, \phi_L(\cdot)]$ of degree f to share $\boldsymbol{z}$.
  - $\boldsymbol{b}^* = [b_1^*, \ldots, b_n^*]$ where $b_k^* = 1$ for all $k \in [n]$.

  Then, send to the adversary, for each corrupt node i, $(\boldsymbol{s}_i, \boldsymbol{w}_i, \boldsymbol{v})$ and degree-proof π_v where for each $k \in [L]$, $(s_{i,k}, w_{i,k}, v_k)$ is a share for i over $\phi_k(\cdot)$.
- If it receives $\boldsymbol{\phi} = [\phi_1(\cdot), \ldots, \phi_L(\cdot)]$ and $\boldsymbol{b}^* \in \{0, 1, \perp\}^n$ from a corrupt dealer in any round, check that
  - $\phi_k(\cdot)$ is a degree-f polynomial for all $k \in [L]$, and
  - If there is an honest node i with $b_i^* = 1$, then for all honest node $k \in [n]$, $b_k^* \neq \perp$.

  Otherwise, set $\boldsymbol{\phi}, \boldsymbol{b}^* = \perp$.
- In round 7, send $(\boldsymbol{s}_i, \boldsymbol{w}_i, \boldsymbol{v})$ and $b = b_i^*$ to each honest node $i \in [n]$ if $b_i \neq \perp$ where for each $k \in [L]$, $(s_{i,k}, w_{i,k}, v_k)$ is a share for node i over $\phi_k(\cdot)$. For node i with $b_i^* = \perp$, send $b = 0$ as the success bit.

Figure 3: Ideal functionality of our VSS protocol

guaranteed output of VSS. If $b_i^* = 1$ for any honest node i, then for all $k \in [n]$, $b_k^* \neq \perp$. Thus, $\mathcal{F}_{\mathsf{VSS}}$ computes the shares for honest nodes over the polynomials $\boldsymbol{\phi}$ sent by $\mathcal{S}$, hence the same output in the $\mathcal{S}$'s local execution. Corrupt nodes' outputs and the message sent directly from $\mathcal{S}$ are both exactly the same as the local execution.

Honest dealer case. $\mathcal{S}$ locally executes VSS with $\mathcal{A}$ except that honest nodes deviate from the protocol as follows:

(1) The honest dealer performs honest sharing for corrupt nodes, but for honest nodes, perform sharing with fake secrets. Specifically, let $(\hat{\boldsymbol{s}}_i, \hat{\boldsymbol{w}}_i, \hat{\boldsymbol{v}})$ be the share for each corrupt node i and $\hat{\pi}_v$ be the degree proof, which $\mathcal{S}$ receives from $\mathcal{F}_{\mathsf{VSS}}$. For each $k \in [L]$, the dealer computes the sharing polynomial $\phi_k(\cdot)$ (for each $k \in [L]$) by interpolation using $\phi_k(i) = \hat{s}_{i,k}$ for all $i \in [f]$ and a randomly chosen $\phi_k(f+1)$. The witness $w_{i,k}$ is set to $\hat{w}_{i,k}$ for each corrupt i. The commitment vector is $\boldsymbol{v} = \hat{\boldsymbol{v}}$. The degree proof for the sharing polynomials is $\pi_v = \hat{\pi}_v$. Except for these, the dealer behaves as specified.
(2) Each honest node i will not send accuse_i. Namely, the honest nodes, despite receiving invalid shares, behave as if the sharing was successful.

In both the ideal world and the real world, nodes' outputs are shares of $\boldsymbol{z}$ over randomly sampled polynomials $\boldsymbol{\phi}$ and are identically distributed in both. The rest of the proof shows that (for each sampled $\boldsymbol{\phi}$) what $\mathcal{Z}$ receives from $\mathcal{S}$ in the ideal world is indistinguishable from what is received from $\mathcal{A}$ in the real world. It is easy to see that what $\mathcal{Z}$ receives is identical in both worlds except for variables dependent on the encoding polynomials $\psi_i(\cdot)$ and $\psi_i'(\cdot)$ for each honest i. The encoding polynomial $\psi_i(\cdot)$ is interpolated from the independently randomized symbols $\phi_k(i) \oplus \rho_{i,k}$. Since the PRF key sk_i is unknown to $\mathcal{Z}$, each of these symbols and hence the encoding polynomial $\psi_i(\cdot)$ is indistinguishable in both worlds, The same argument holds for $\psi_i'(\cdot)$. Therefore, what $\mathcal{Z}$ receives is indistinguishable in both worlds. □

5.4 Reducing Computational Overhead

As mentioned, our protocol follows the construction of hbACSS [41]. The main distinction from hbACSS lies in the use of our augmented KZG commitment with additional degree proof. An important point to discuss is how much overhead is introduced by the degree proof. One can easily observe that the communication cost is negligible, as the degree proof is a single Feldman commitment of size $O(d)$. The dealer's computational overhead for generating degree proof is also trivial, requiring only $O(d)$ elliptic curve group exponentiation. In comparison, computing commitment and witnesses requires $O(dLn)$ group exponentiation. However, verifying a degree proof requires L paring operations, which introduces an overhead comparable to the cost of verifying evaluation proofs. This issue can be partially mitigated by using another commitment scheme in place of the Feldman commitment.

Degree proof from masked polynomial. Essentially, we can adopt any aggregatable commitment for degree proof. One such example is the commitment scheme based on random polynomial masking [17, 38]. Specifically, a dealer chooses a random mask polynomial $\phi_0(\cdot)$ besides the sharing polynomials $\phi_1(\cdot), \ldots, \phi_L(\cdot)$, and open a random linear combination

$$\Phi(\cdot) = \sum_{0 \leq k \leq L} \rho_k \cdot \phi_k(\cdot)$$

as the proof of degree, where ρ_k is random values chosen independently from the choice of polynomials. The random polynomial $\phi_0(\cdot)$ is also shared among nodes (i.e., node i receives $\phi_0(i)$) along with the KZG commitment $v_0 = g^{\phi_0(\tau)}$ and associated evaluation proofs.

Each shareholder i performs the verification similar to that of Figure 1. Specifically, node i checks that

$$\Phi(i) = \sum_{0 \leq k \leq L} \rho_k \cdot \phi_k(i)$$
$$g^{\Phi(\tau)} = \prod_{0 \leq k \leq L} v_k^{\rho_k}$$

The proof of degree binding for the Feldman variant (Lemma 4.2) can be generalized to the above variant. In essence, the proof shows that the committed polynomials on the Feldman commitments and the KZG commitments are equal. Since the aggregated commitment is degree binding by design, this implies that the polynomials binding to the KZG commitments also have the claimed degree. The equality of polynomials follows from the fact that the adversary cannot choose different polynomials whose random combinations agree. These arguments directly apply to the above approach.

Finally, to briefly analyze the overhead for each shareholder, the KZG verification takes 3 pairings per secret while the verification of degree proof takes one group exponentiation per secret. Therefore, we believe this is a worthwhile cost for achieving security under the standard model and for providing reconfiguration friendliness.

5.5 Extension to Asynchronous VSS

We have presented our VSS protocol in the synchrony model. The protocol can be extended to support an asynchronous network using existing techniques. In an asynchronous network, there is no bound on the message delivery delay. Thus, asynchronous VSS (AVSS) allows each node to output its share whenever it receives enough messages. Also, nodes do not output the success bit anymore because failure to receive a share at some point does not mean a node will never receive its share. The *guaranteed output* property is changed accordingly, which is also called *completeness*.

- If an honest node i outputs the share s_i, then every honest node j eventually outputs the share s_j.

To achieve this property, we have to make two major modifications to our synchronous VSS.

Timed algorithm to event-triggered. The first standard modification is to make the algorithm event-triggered and non-blocking. Each event must be upon receiving enough valid messages as messages are not guaranteed to be timely delivered in an asynchronous network. For example, forwarding code words (tagged Forward) happens upon receiving all code words that are verified with the commitments and degree proofs. Similarly, sending "ready" happens upon receiving all valid shares and forwarding all valid code words, and sending "complete" happens upon receiving $2f + 1$ "ready". Gradecast should be replaced by reliable broadcast, which allows nodes to receive the sender's message at any time.

Share recovery for completeness. The key challenge in extending synchronous to asynchronous VSS is to achieve *completeness* stated above. In synchronous VSS, once a valid accusation is received, honest nodes can simply abort the sharing by outputting the failure bit $b = 0$. Under asynchrony, however, an honest node's accusation can be delayed arbitrarily, which can be after other honest nodes output their shares. Therefore, the honest node that failed to receive its valid share needs a way to recover its share. This problem is solved by the ShareRecovery algorithm (Algorithm 2) of hbAVSS [41]). In a nutshell, the missing shares can be interpolated from $f + 1$ honest nodes' shares.

6 RELATED WORKS

Verifiable secret-sharing [15, 35] is an essential tool in threshold cryptography. There has been a lot of research on single-secret VSS for different network models and different security levels. For a brief review, existing information-theoretic VSS protocols tolerate $f < n/3$ corruption and cost $O(\kappa n^3)$ communication both in synchrony and asynchrony [16]. With computational security, a synchronous VSS tolerates $f < n/2$ corruption and cost $O(\kappa n^2)$ communication [8], and an asynchronous AVSS tolerates $f < n/3$ corruption and cost $O(\kappa n^2)$ communication [3]. Single-secret VSS has applications including distributed key generation for threshold signature/encryption [24] and randomness beacons [8, 18]. In some applications, however, a dealer must share multiple secrets [6, 42]. Such a multi-secret VSS can be solved more efficiently by amortizing some costs over multiple secrets. Here, we review several existing approaches to achieve multi-secret VSS with linear cost. Along the way, we also review the polynomial commitments used in these schemes.

Player-elimination in MPC. The preprocessing phase of MPC usually involves each node sharing a large number of secrets, typically proportional to the number of gates in the circuit. Thus, amortizing the sharing cost over multiple secrets is a natural problem in MPC [5, 17, 25]. The common approach is to detect corrupt nodes and eliminate them from the execution. The cost of eliminating at most f corrupt nodes is amortized over the remaining honest execution with linear communication. However, this approach takes $O(n)$ rounds and does not work in asynchrony. We also note that it is unclear if this approach is applicable to standalone VSS.

Packed secret-sharing. Patra et al. [34] presents asynchronous multi-secret VSS with $O(\kappa Ln + n^2)$ communication but tolerates $f < n/4$ corruption. Their protocol takes the classic packed sharing approach, namely a linear number of secrets are shared with a single polynomial. However, this approach inherently sacrifices the corruption threshold.

IDA-then-accuse. Another natural approach, which we follow to some extent, is to rely on an information dispersal algorithm (IDA). Yurek et al. [41] presented an asynchronous multi-secret VSS with linear communication. The protocol uses a black-box IDA to disseminate a batch of shares to each node. Nodes then validate the shares and implicate the dealer upon receiving incorrect shares. Since the implication is required to contain all the $L = O(n)$ received shares, the total communication cost is at least $\Omega(n^3)$. hbACSS has several variants depending on the polynomial commitment scheme used. hbACSS0 and hbACSS2 use a polynomial commitment based on Bulletproofs [11] and does not need the powers of tau setup. But the communication cost is $\Omega(\kappa Ln \log n + \kappa n^3)$ and the commitment is not homomorphic. hbACSS1 achieves $O(\kappa Ln + \kappa n^3)$ using the original KZG commitment.

Bingo. The closest work to this paper is Bingo [2], an asynchronous multi-secret VSS with optimal $O(\kappa Ln + \kappa n^2)$ communication. They use a bivariate polynomial to share multiple secrets at once with amortized linear cost per secret. They use the KZG commitment extended for bivariate polynomials.

The main advantage of this work over Bingo is that we achieve degree binding without idealized group assumptions. Bingo adopts the original KZG commitment and proves it is degree-binding (they call it *interpolation binding*) under the Algebraic adversary (i.e., in the AGM). We have eliminated the idealized group assumption of AGM by separately detecting the high-degree polynomials using an aggregation of linear-sized commitments as degree proof.

Bingo achieves adaptive security, while our scheme only achieves static security. Achieving adaptive security with our augmentation is interesting future work.

Optimistic approach. Optimizing the cost in an optimistic execution is another common approach. Basu et al. [4] present a single-secret VSS protocol with linear communication in failure-free cases. However, the protocol incurs quadratic communication in the worst case. The protocol uses the original KZG commitment. Benhamouda et al. [7] present a synchronous multi-secret VSS protocol that costs amortized linear communication when the dealer is honest. When the dealer is malicious, honest nodes' shares are opened to everybody, thus costing quadratic communication per secret.

7 DISCUSSION AND CONCLUSION

In this work, we have revisited the security of the KZG commitment in the use of VSS and pointed out two issues with the original KZG commitment: 1) it is not proven degree-binding without idealized group assumptions, and 2) it does not support multiple degrees with a single setup. We have augmented the KZG commitment to make it degree-binding and presented a multi-secret VSS protocol building on our extended KZG commitment. Finally, we discuss some limitations of our protocols to conclude the paper.

Fault-tolerance limit. Another important question is whether it is possible to achieve linear communication with optimal corruption threshold $f < n/2$ in synchrony. The bottleneck is in disseminating shares. The classic technique of IDA using erasure coding [12] (which we also adopt) has a fundamental limit on the threshold. Specifically, nodes can expect to receive votes from $n - f$ nodes, of which $n - 2f$ are honest and forward the assigned code words that help shareholders recover the original message. We must have $n - 2f = \Omega(n)$ to achieve linear communication with this approach.

Computational complexity. Another general problem with the use of KZG commitment (both in the original and ours) is its computation cost. Computing a witness for each share requires a linear number of group exponentiation. Some previous works showed how to compute the witnesses with logarithmic computation cost in a batch setting [39, 41]. However, their experimental results showed that it is still costly compared to linear-sized commitment schemes. So the computation/communication trade-off still exists.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers at ACM CCS 2023 for their helpful feedback. We thank Dahlia Malkhi and Andrew Miller for valuable discussions related to this paper. This work is funded in part by the NSF award 2240976.

REFERENCES

[1] Ittai Abraham, Srinivas Devadas, Danny Dolev, Kartik Nayak, and Ling Ren. 2019. Synchronous Byzantine Agreement with Expected $O(1)$ Rounds, Expected $O(n^2)$ Communication, and Optimal Resilience. In *Financial Cryptography and Data Security (FC)*. Springer, 320–334.

[2] Ittai Abraham, Philipp Jovanovic, Mary Maller, Sarah Meiklejohn, and Gilad Stern. 2022. Bingo: Adaptively Secure Packed Asynchronous Verifiable Secret Sharing and Asynchronous Distributed Key Generation. *IACR Cryptology ePrint Archive, Report 2022/1759* (2022).

[3] Nicolas Alhaddad, Mayank Varia, and Haibin Zhang. 2021. High-threshold avss with optimal communication complexity. In *Financial Cryptography and Data Security (FC)*. Springer, 479–498.

[4] Soumya Basu, Alin Tomescu, Ittai Abraham, Dahlia Malkhi, Michael K Reiter, and Emin Gün Sirer. 2019. Efficient verifiable secret sharing with share recovery in BFT protocols. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2387–2402.

[5] Zuzana Beerliová-Trubíniová and Martin Hirt. 2008. Perfectly-secure MPC with linear communication complexity. In *Theory of Cryptography Conference (TCC)*. Springer, 213–230.

[6] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. 1988. Completeness theorems for noncryptographic fault-tolerant distributed computations. In *Annual ACM Symposium on Theory of Computing (STOC)*. 1–10.

[7] Fabrice Benhamouda, Shai Halevi, Hugo Krawczyk, Alex Miao, and Tal Rabin. 2022. Threshold Cryptography as a Service (in the Multiserver and YOSO Models). In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 323–336.

[8] Adithya Bhat, Nibesh Shrestha, Zhongtang Luo, Aniket Kate, and Kartik Nayak. 2021. Randpiper–reconfiguration-friendly random beacons with quadratic communication. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 3502–3524.

[9] Dan Boneh and Xavier Boyen. 2008. Short signatures without random oracles and the SDH assumption in bilinear groups. *Journal of Cryptology* 21, 2 (2008), 149–177.

[10] Dan Boneh, Ben Lynn, and Hovav Shacham. 2001. Short signatures from the Weil pairing. In *Annual International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT)*. Springer, 514–532.

[11] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. 2018. Bulletproofs: Short proofs for confidential transactions and more. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 315–334.

[12] Christian Cachin and Stefano Tessaro. 2005. Asynchronous verifiable information dispersal. In *IEEE Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 191–201.

[13] Ran Canetti. 2001. Universally composable security: A new paradigm for cryptographic protocols. In *Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 136–145.

[14] Ran Canetti, Asaf Cohen, and Yehuda Lindell. 2015. A simpler variant of universally composable security for standard multiparty computation. In *Annual International Cryptology Conference (CRYPTO)*. Springer, 3–22.

[15] Benny Chor, Shafi Goldwasser, Silvio Micali, and Baruch Awerbuch. 1985. Verifiable secret sharing and achieving simultaneity in the presence of faults. In *Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 383–395.

[16] Ashish Choudhury. 2020. Optimally-resilient unconditionally-secure asynchronous multi-party computation revisited. *IACR Cryptology ePrint Archive, Report 2020/906* (2020).

[17] Ivan Damgård and Jesper Buus Nielsen. 2007. Scalable and unconditionally secure multiparty computation. In *Annual International Cryptology Conference (CRYPTO)*. Springer, 572–590.

[18] Sourav Das, Vinith Krishnan, Irene Miriam Isaac, and Ling Ren. 2022. Spurt: Scalable distributed randomness beacon with transparent setup. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2502–2517.

[19] Sourav Das, Thomas Yurek, Zhuolun Xiang, Andrew Miller, Lefteris Kokoris-Kogias, and Ling Ren. 2022. Practical asynchronous distributed key generation. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2518–2534.

[20] Danny Dolev and H. Raymond Strong. 1983. Authenticated algorithms for Byzantine agreement. *SIAM J. Comput.* 12, 4 (1983), 656–666.

[21] Paul Feldman. 1987. A practical scheme for non-interactive verifiable secret sharing. In *Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 427–438.

[22] Amos Fiat and Adi Shamir. 1987. How to prove yourself: Practical solutions to identification and signature problems. In *Annual International Cryptology Conference (CRYPTO)*. Springer, 186–194.

[23] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. 2018. The algebraic group model and its applications. In *Annual International Cryptology Conference (CRYPTO)*. Springer, 33–62.

[24] Rosario Gennaro, Stanisław Jarecki, Hugo Krawczyk, and Tal Rabin. 1999. Secure distributed key generation for discrete-log based cryptosystems. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*. Springer, 295–310.

[25] Martin Hirt and Jesper Buus Nielsen. 2006. Robust multiparty computation with linear communication complexity. In *Annual International Cryptology Conference (CRYPTO)*. Springer, 463–482.

[26] Bin Hu, Zongyang Zhang, Han Chen, You Zhou, Huazu Jiang, and Jianwei Liu. 2022. DyCAPS: Asynchronous Proactive Secret Sharing for Dynamic Committees. *IACR Cryptology ePrint Archive, Report 2022/1169* (2022).

[27] Aniket Kate, Gregory M Zaverucha, and Ian Goldberg. 2010. Constant-size commitments to polynomials and their applications. In *Annual International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT)*. Springer, 177–194.

[28] Aniket Kate, Gregory M Zaverucha, and Ian Goldberg. 2010. Polynomial Commitments. (2010). https://cacr.uwaterloo.ca/techreports/2010/cacr2010-10.pdf

[29] Jonathan Katz and Chiu-Yuen Koo. 2009. On expected constant-round protocols for byzantine agreement. *J. Comput. System Sci.* 75, 2 (2009), 91–112.

[30] Leslie Lamport, Robert Shostak, and Marshall Pease. 1982. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems* 4, 3 (1982), 382–401.

[31] Donghang Lu, Thomas Yurek, Samarth Kulshreshtha, Rahul Govind, Aniket Kate, and Andrew Miller. 2019. Honeybadgermpc and asynchromix: Practical asynchronous mpc and its application to anonymous communication. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 887–903.

[32] Sai Krishna Deepak Maram, Fan Zhang, Lun Wang, Andrew Low, Yupeng Zhang, Ari Juels, and Dawn Song. 2019. CHURP: dynamic-committee proactive secret sharing. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*. 2369–2386.

[33] Achour Mostéfaoui, Hamouma Moumen, and Michel Raynal. 2014. Signature-free asynchronous Byzantine consensus with $t < n/3$ and $O(n^2)$ messages. In *ACM Symposium on Principles of Distributed Computing (PODC)*. 2–9.

[34] Arpita Patra, Ashish Choudhury, and C Pandu Rangan. 2010. Communication efficient perfectly secure VSS and MPC in asynchronous networks with optimal resilience. In *International Conference on Cryptology in Africa (AFRICACRYPT)*. Springer, 184–202.
[35] Torben Pryds Pedersen. 2001. Non-interactive and information-theoretic secure verifiable secret sharing. In *Annual International Cryptology Conference (CRYPTO)*. Springer, 129–140.
[36] Michael O Rabin. 1989. Efficient dispersal of information for security, load balancing, and fault tolerance. *Journal of the ACM (JACM)* 36, 2 (1989), 335–348.
[37] Irving S Reed and Gustave Solomon. 1960. Polynomial codes over certain finite fields. *Journal of the society for industrial and applied mathematics* 8, 2 (1960), 300–304.
[38] Victor Shoup and Nigel P Smart. 2023. Lightweight Asynchronous Verifiable Secret Sharing with Optimal Resilience. *IACR Cryptology ePrint Archive, Report 2023/536* (2023).
[39] Alin Tomescu, Robert Chen, Yiming Zheng, Ittai Abraham, Benny Pinkas, Guy Golan Gueta, and Srinivas Devadas. 2020. Towards scalable threshold cryptosystems. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 877–893.
[40] Robin Vassantlal, Eduardo Alchieri, Bernardo Ferreira, and Alysson Bessani. 2022. Cobra: Dynamic proactive secret sharing for confidential bft services. In *IEEE Symposium on Security and Privacy (S&P)*. IEEE, 1335–1353.
[41] Thomas Yurek, Licheng Luo, Jaiden Fairoze, Aniket Kate, and Andrew Miller. 2022. hbACSS: How to Robustly Share Many Secrets. In *Network and Distributed System Security Symposium (NDSS)*.
[42] Thomas Yurek, Zhuolun Xiang, Yu Xia, and Andrew Miller. 2022. Long live the honey badger: Robust asynchronous dpss and its applications. *IACR Cryptology ePrint Archive, Report 2022/971*.

A GRADE-CAST FOR A LARGE MESSAGE.

This section presents a concrete protocol for gradecast, which takes 3 rounds and costs $O(Ln + \kappa n^2)$ communication to send a message M of size L. The protocol is described in Algorithm 4.

Error correcting code. Our protocol uses Reed-Solomon codes for error-correcting codes. We review the interface and the property here. The encoding algorithm takes the message M consisting of k symbols and outputs m code words:

$$c_1, .., c_m \leftarrow \mathsf{RS.Enc}(M, m, k)$$

The decoding algorithm takes a set X of code words with at most r errors and decodes the original M with k symbols:

$$M \leftarrow \mathsf{RS.Dec}(k, r, X)$$

It is well-known that the decoding algorithm can correct up to r errors if $|X| \geq k + 2r$ [37].

Correctness. We show the protocol GC satisfies *graded agreement* and *consistency* below (*validity* is straightforward).

Lemma A.1 (Graded agreement). *If an honest node outputs* $(M, 1)$*, then all honest nodes output* $(M, *)$.

Proof. An honest node outputs a message M with grade $b = 1$ when the node receives $\{\text{"vote"}, \mathsf{H}(M)\}$ from $2f + 1$ nodes. At least $f + 1$ of these must be received by all honest nodes, thus they all have at least grade $b \geq 0$. So, the rest of the proof shows all honest nodes can reconstruct the message M.

Among the $2f + 1$ nodes who sent $\{\text{"vote"}, \mathsf{H}(M)\}$, there is a set H of at $f + 1$ honest nodes. Each honest node $i \in H$ must have received the same pair of a code word c_i and the hash $\mathsf{H}(M)$ from at least $2f + 1$ nodes, out of which at least $f + 1$ are honest. The $f + 1$ honest nodes who sent $\{c_i, \mathsf{H}(M)\}$ must have received from the dealer the same message M, whose hash matches $h = \mathsf{H}(M)$, and sent every code word c_j to the assigned node j, which is then forwarded by the node j if honest. Therefore each honest node receives a set X of at least $2f + 1$ code words of M with at most $r = |X| - (2f + 1)$ possible error, thus reconstructing the message M. □

GC – Grade-cast

A dealer $\mathcal{D}$ has a message M to send to all nodes.

- *Round 1.* The dealer $\mathcal{D}$ sends M to all nodes.
- *Round 2.* If node i receives a single message M from the dealer, then generates
  $$c_1, .., c_n \leftarrow \mathsf{RS.Enc}(M, n, f + 1)$$
  and send $\{c_j, \mathsf{H}(M)\}$ to each node j.
- *Round 3.* If node i receives $\{c, h\}$ from $m > f$ nodes, then i forwards it to all nodes in R. If $m > 2f$, then i also sends $\{\text{"vote"}, h\}$ to all nodes.
- *At the end of round 3.* If node i receives $\{\text{"vote"}, h\}$ from $m > 2f$ nodes, then node i sets $b = 1$. If $f < m \leq 2f$ then $b = 0$. If $m > f$, reconstruct M by invoking
  $$M \leftarrow \mathsf{RS.Dec}(f + 1, r, X)$$
  where X is the set of more than $2f$ code words received with the same hash, and $r = |X| - (2f + 1)$. If $\mathsf{H}(M) = h$, then i outputs M with grade b.

Figure 4: Gradecast with linear communication.

Lemma A.2 (Consistency). *If two honest nodes output* $(M, *)$ *and* $(M', *)$*, then* $M = M'$.

Proof. Suppose for contradiction, two honest nodes output different messages M and M'. An honest node outputs a message M when the node receives $\{\text{"vote"}, \mathsf{H}(M)\}$ from at least $f + 1$ nodes, out of which at least one node must be honest. The honest node who sent $\{\text{"vote"}, \mathsf{H}(M)\}$ must have received the same code word c and the hash $\mathsf{H}(M)$ from a set T of at least $2f + 1$ nodes. By the same logic, there is a set T' of $2f + 1$ nodes who sent the hash $\mathsf{H}(M')$ along with the codes of M'. The two sets T and T' have at least $f + 1$ intersection, out of which at least one node must be honest. However, an honest node sends a hash only if it receives a single message. Therefore, an honest node could not have sent both $\mathsf{H}(M)$ and $\mathsf{H}(M')$; a contradiction. □

B DEGREE BINDING AND PDH ASSUMPTION

In §3, we have explained the *strong correctness* property [28] proven under the PDH assumption is insufficient and provided a degree binding property as a sufficient alternative. We observe that with our definition of degree binding, the proof based on the PDH assumption does not apply. To clarify, let us review the definition of the PDH assumption below (refer to Definition 2 in [27]).

Definition B.1 (t-PDH Assumption.). Let $\tau \in \mathbb{Z}_q^*$ be a randomly chosen value. For any PPT adversary $\mathcal{A}$, the probability

$$\Pr[\mathcal{A}([g, g^{\tau}, .., g^{\tau^t}]) \rightarrow (\phi(\cdot), g^{\phi(\tau)})]$$

is negligible for any polynomial $\phi(\cdot) \in \mathbb{Z}_q[x]$ s.t. the degree d satisfies $t < d < 2^{\kappa}$.

The above definition says the PDH problem (i.e., computing the polynomial $\phi(\cdot)$ and the commitment $g^{\phi(\tau)}$) is hard to solve for any degree d. To carefully look into the definition, when the degree is of super polynomial, i.e., $d = \omega(\text{poly}(\kappa))$, the statement is trivially true since the representation of the solution $\phi(\cdot)$ is of super polynomial-sized. However, it does not imply the hardness of computing a moderate number (i.e., polynomial in κ) of evaluations on the polynomial $\phi(\cdot)$ of super-polynomial degree, which must be also excluded with our degree binding. Therefore, the PDH assumption does not apply to the proof of degree binding.

C DPSS WITH CONSTANT-ROUND EPOCH

In this section, we design a dynamic-committee proactive secret-sharing (DPSS) building on our multi-secret VSS. Our protocol always completes the secret handover in $O(1)$ round and costs $O(\kappa n^3)$ communication. The reconfiguration-friendly nature of our augmented KZG commitment allows our protocol to re-size the committee over repeated handovers without redoing the powers of tau setup.

Definition. DPSS [32] allows one committee to hand over a shared secret to another committee. The repetition of the handover allows the system to keep the shared secret available while changing the responsible members. The classic definition of proactive secret-sharing assumes each committee shares a secret over a unique polynomial. Specifically, each node i in the old committee R holds a share $s_i = \phi(i)$ over a single polynomial $\phi(\cdot)$, and the handover protocol allows each node j in the new committee R' to receive a share $s_j = \phi'(j)$ over a new polynomial $\phi'(\cdot)$ sharing the same secret $\phi'(0) = \phi(0)$. However, this definition implicitly assumes consensus on the unique random polynomial $\phi'(\cdot)$. This makes all existing DPSS protocols [26, 32, 40, 42] incur $\Omega(n)$ or $\Omega(\kappa)$ round to complete the handover in the worst case, during which the committee must stay online. To avoid this, we let a committee hold multiple polynomials (sharing the same secret). We give a more formal definition below.

Let R and R' be two (possibly joint) sets of n nodes, where at most $f < n/3$ nodes are corrupt in each committee. Each node $i \in R$ inputs a vector $\boldsymbol{s}_i = [\phi_1(i), .., \phi_n(i)]$ of n shares where $\phi_1(\cdot), .., \phi_n(\cdot)$ are polynomials of degree f and share the same secret (i.e., there is a unique secret z and $\phi_k(0) = z$ for all $k \in [n]$). We also allow some node i to include an empty share $s_{i,k} = \perp$ for some entry $k \in [n]$. After running the handover protocol, each node $j \in R'$ outputs a vector $\boldsymbol{s}_j = [\phi'_1(i), .., \phi'_n(i)]$ of n shares over new random polynomials $\phi'_1(\cdot), .., \phi'_n(\cdot)$ that shares the same secret z. To make sure an available (i.e., recoverable by honest nodes) secret in R is always available in committee R', we also require the following property.

- If there exists an entry $k \in [n]$ s.t. all honest node $i \in R$ holds $s_{i,k} \neq \perp$, then $s_{j,l} \neq \perp$ for all honest $j, l \in R'$.

We also assume each node i when it inputs(outputs) a share $\phi_k(i)$ also inputs(outputs) the KZG witness and the commitment.

Resizable committee. As mentioned, our DPSS protocol can easily support changing committee size and the corruption threshold over repeated handovers. Specifically, our protocol can support any number of nodes $0 < n \leq m$ in each committee (as long as up to $f < n/3$ are corrupt) under the availability of a structured reference string $[g, g^{\tau}, .., g^{\tau^m}]$ for a random $\tau \in \mathbb{Z}_q^*$. Therefore, it is also allowed that R and R' have different numbers of nodes. For simplicity of presentation, we assume they are both n nodes.

Empty share. In our DPSS, it is possible that an honest node's share $\boldsymbol{s}_i$ includes an empty share $s_{i,k} = \perp$ for some entry $k \in [n]$. Looking ahead, this results in an honest node sharing an empty value $\perp$. For technical reasons, we consider $\perp$ as a share over an *empty polynomial* for any index $i \in [n]$.

Lagrange coefficient. We use the notation λ_k to denote the Lagrange coefficient for index k to interpolate a polynomial evaluation on index 0. Specifically, for a set T of indices (used for interpolation), the Lagrange coefficient for index $k \in T$ is described as

$$\lambda_k = \prod_{i \in T \setminus k} \frac{i}{i-k}.$$

We omit to mention T when it is clear from the context.

C.1 Our Protocol

Our DPSS protocol is described in Figure 5. We elaborate on the intuition below.

DPSS from leader-aided handover. For ease of understanding, let us first consider a solution to the classic DPSS definition where each committee holds a single polynomial. Assuming a consensus protocol, we can design a simple handover protocol using the classic re-sharing share technique [6]. Specifically, each member $i \in R$ of the old committee re-shares its share $s_i = \phi(i)$ by invoking VSS (let us denote VSS_i). Let $T \subset R$ be an agreed-on set of $f+1$ nodes whose VSS were successful. Each member $j \in R'$ of the new committee can compute its new share s_j as follows

$$s_j = \phi'(j) = \sum_{i \in T} \lambda_i \cdot \psi_i(j)$$

where $\psi_i(j)$ is the share received from VSS_i. The secret $\phi'(0)$ shared over the new polynomial is an interpolation from $\psi_i(0) = \phi(i)$ for all $i \in T$, thus matches the original secret $\phi(0)$. All of the coefficients of $\phi'(\cdot)$ except for dimension zero are defined by the coefficients of the $f+1$ polynomials, which include at least one random polynomial chosen by an honest node. Thus an adversary cannot determine honest nodes' shares. The communication cost is $O(\kappa n^3)$ as we have n single-secret VSSs each of them costs at least $O(n^2)$ communication.

Back to our problem with the constant-round limit, we cannot agree on any single bit. Since we have exponentially many possible choices of T, nodes cannot compute shares over an agree-on polynomial. To address this issue, we let each member $i \in R'$ of the new committee serve as a leader and choose its own set T_i. This results in a committee holding n polynomials defined by $T_1, .., T_n$. Now, since each node $k \in R$ inputs n shares, the leader i also designates the index l. Then, the i-th share of node $j \in R'$ is computed as

$$s_{j,i} = \sum_{k \in T_i} \lambda_k \cdot \psi_{k,l}(j).$$

where $\psi_{k,l}(\cdot)$ is the polynomial used for l-th sharing of VSS_k (i.e., for sharing $\phi_l(i)$). Here, since we have n parallel handovers in each epoch, we have n^2 total secret-sharing. However, as each node has

Each node $i \in R$ inputs $(\boldsymbol{s}_i, \boldsymbol{w}_i, \boldsymbol{v})$.

// Round 1–7

Re-share. Each node $i \in R$ starts re-sharing its share $\boldsymbol{s}_i$ with the new committee R' by invoking VSS_i.

Share-proof. Let $\psi_{i,k}(\cdot)$ be the polynomial used to share $s_{i,k}$ in VSS_i and $\eta_{i,k}$ is the witness for $\psi_{i,k}(0)$. Node i also sends to all of the members of the new committee R'

$$\text{share-proof}_i := (\boldsymbol{v}, \boldsymbol{w}_i, \boldsymbol{\mu}_i, \boldsymbol{\eta}_i)$$

by invoking gradecast GC_i, where $\boldsymbol{\mu}_i = [g^{s_{i,1}}, .., g^{s_{i,n}}]$ and $\boldsymbol{\eta}_i = [\eta_{i,1}, .., \eta_{i,n}]$.

// Round 8–10.

Verify share-proof. Each node $j \in R'$ verifies share-proof$_i$ received from GC_i by checking that for all $k \in [n]$, the committed share $s_{i,k}$ (i.e., $\mu_{i,k} = g^{s_{i,k}}$) satisfies

$$\mathsf{VerifyEval}(v_k, i, s_{i,k}, w_{i,k}) = 1$$
$$\mathsf{VerifyEval}(u_{i,k}, 0, s_{i,k}, \eta_{i,k}) = 1$$

where $u_{i,k}$ is the commitment to $\psi_{i,k}(\cdot)$ received during VSS_i.

Choose polynomials. Each node $j \in R'$ chooses a set of $f+1$ indices $T_j \subset [n]$ and an index $l \in [n]$ that satisfies both of the following.

(1) For all $k \in T_j$, VSS_k has outputted $(\hat{\boldsymbol{s}}_{j,k}, \hat{\boldsymbol{w}}_{j,k}, \hat{\boldsymbol{v}}_k)$ with success bit $b = 1$ s.t. the $(\hat{s}_{j,k,l}, \hat{w}_{j,k,l}, \hat{v}_{k,l}) \neq \perp$ (i.e., the l-th share is not a share over an empty polynomial).
(2) There is a commitment v, and for all $i \in T_j$, GC_i has outputted with grade $b = 1$ a verified share-proof$_i$ that includes $\boldsymbol{v} = [.., v_l = v, ..]$ (i.e., the l-th commitment is v).

then, sends (T_j, l) to all members of the new committee R' by invoking a gradecast GC'_j.

// At the end of round 10.

New shares. Each node $j \in R'$ verifies, for each $i \in [n]$, the output (T_i, l) from GC'_i by checking that

(1) For all $k \in T_i$, VSS_k has outputted a share $(\hat{\boldsymbol{s}}_{j,k}, \hat{\boldsymbol{w}}_{j,k}, \hat{\boldsymbol{v}}_k)$ s.t. $(\hat{s}_{j,k,l}, \hat{w}_{j,k,l}, \hat{v}_{k,l}) \neq \perp$.
(2) There is a commitment v, and for all $k \in T_i$, share-proof$_k$ is verified and includes $\boldsymbol{v} = [.., v_{k,l} = v, ..]$.

Then, computes the new shares $\boldsymbol{s}_j = [s_{j,1}, .., s_{j,n}]$, the commitments $\boldsymbol{v} = [v_1, .., v_n]$ and the witnesses $\boldsymbol{w}_j = [w_{j,1}, .., w_{j,n}]$. For $i \in [n]$ with verified $\{T_i, l\}$,

$$s_{j,i} = \sum_{k \in T_i} \lambda_k \cdot \hat{s}_{j,k,l}, \quad v_i = \prod_{k \in T_i} \hat{v}_{k,l}^{\lambda_k}, \quad w_{j,i} = \prod_{k \in T_i} \hat{w}_{j,k,l}^{\lambda_k}$$

otherwise $s_{j,i}, v_i, w_{j,i} \leftarrow \perp$.

Node j outputs the new share $(\boldsymbol{s}_j, \boldsymbol{w}_j, \boldsymbol{v})$

Figure 5: Our DPSS protocol with constant-round handover.

$L = n$ shares to re-share, we can use our multi-secret VSS protocol with $O(\kappa n^2)$ total communication, thus the overall communication cost is still $O(\kappa n^3)$.

Validate shares. When a node $i \in R$ re-shares its share $s_{i,k} = \phi_k(i)$, it must prove that the shared secret $\psi_{i,k}(0)$ (in VSS_i) matches the original share $\phi_k(i)$. To this end, the node i also sends (via a gradecast) share-proof$_i$ that contains a committed share $\mu_{i,k} = g^{s_{i,k}}$, the witness $\eta_{i,k}$ for $\psi_{i,k}(0)$, as well as the commitment v_k to $\phi_k(\cdot)$ and the witness $w_{i,k}$ for $\phi_k(i)$. Each of the new committee members verifies the share-proof$_i$ by checking that

$$\mathsf{VerifyEval}(v_k, i, s_{i,k}, w_{i,k}) = 1$$
$$\mathsf{VerifyEval}(u_{i,k}, 0, s_{i,k}, \eta_{i,k}) = 1$$

where $u_{i,k}$ is the commitment to $\psi_{i,k}(\cdot)$ received during VSS_i. Here, note that node j only knows the committed share $g^{s_{i,k}}$ (rather than the share $s_{i,k}$ itself). However, recall that the verification in VerifyEval is in pairing [27]. Thus, it can be done without knowing the $s_{i,k}$. Specifically, checking that $\mathsf{VerifyEval}(v, i, s, w) = 1$ for $\mu = g^s$ can be done by verifying the following equality

$$e(g, v) = e(w, g^{\tau - i}) \cdot e(\mu, g).$$

where e is the pairing.

C.2 Correctness Proof

We prove the correctness of our DPSS protocol. Let z be the secret shared by the old committee R. Namely, $\phi_k(0) = z$ for all $k \in [n]$. We first show that if an honest node $j \in R'$ in the new committee outputs a share $s_{j,k}$, then it is a share $\phi'_k(j)$ over a unique degree-f polynomial $\phi'_k(\cdot)$ of the same secret $\phi_k(0) = z$.

Lemma C.1. *There exists n polynomials $\phi'_1(\cdot), .., \phi'_n(\cdot)$ all with degree-f and $\phi'_k(0) = z$ for all $k \in [n]$ s.t. if an honest node $j \in R'$ outputs a share $\boldsymbol{s}_j$, then $s_{j,k} = \phi'_k(j)$ or $\perp$.*

Proof. Due to the commitment property of VSS, if an honest node $j \in R'$ in the new committee outputs $(\hat{\boldsymbol{s}}_{j,i}, \hat{\boldsymbol{w}}_{j,i}, \hat{\boldsymbol{v}}_i)$ from VSS_i, then for each $k \in [n]$, $(\hat{s}_{j,i,k}, \hat{w}_{j,i,k}, \hat{v}_{i,k})$ is a share for node j over a unique degree-f polynomial $\psi_{i,k}(\cdot)$. Each node $j \in R'$ computes the output $(s_{j,i}, w_{j,i}, v_i)$ for each $i \in [n]$ based on the set T_i of indices and an index $l \in [n]$ proposed by $i \in R$ via GC'_i. Due to the consistency of GC, all honest nodes receive the same (T_i, l). An honest node $j \in R'$ computes the output $(s_{j,i}, w_{j,i}, v_i) \neq \perp$ after verifying that for all $k \in T_i$, share-proof$_k$ is verified and includes the same commitment $v_{k,l} = v$. Since at least one of T_i must be honest, v is the commitment to $\phi_l(\cdot)$. The verification of share-proof$_k = (\boldsymbol{v}, \boldsymbol{w}_k, \boldsymbol{\mu}_k, \boldsymbol{\eta}_k)$ makes sure that $\mu_{k,l} = g^{\phi_l(k)}$ (by the first condition), and further (by the second condition) that $\psi_{k,l}(0) = \phi_l(k)$. Therefore, there is a polynomial uniquely defined by the set of indices T_i

$$\phi'_i(\cdot) = \sum_{k \in T_i} \lambda_k \cdot \psi_{k,l}(\cdot) \qquad (\phi'_i(0) = z)$$

and the new share $(s_{j,i}, w_{j,i}, v_i)$ will be

$$s_{j,i} = \sum_{k \in T_i} \lambda_k \cdot \hat{s}_{j,k,l} = \phi'_i(j)$$

$$v_i = \prod_{k \in T_i} \hat{v}_{k,l}^{\lambda_k} = g^{\phi'_i(\tau)}$$

$$w_{j,i} = \prod_{k \in T_i} \hat{w}_{j,k,l}^{\lambda_k} = g^{\frac{\phi'_i(\tau) - \phi'_i(j)}{\tau - j}}$$

Therefore, if the node j computes a new share $(s_{j,i}, w_{j,i}, v_i) \neq \perp$, then it must be the share on the polynomial $\phi'_i(\cdot)$ uniquely defined by the set T_i. □

Finally, we show that an available secret among R is also always available among R'.

Lemma C.2. *If there exists an entry $k \in [n]$ s.t. all honest node $i \in R$ inputs $s_{i,k} \neq \perp$, then $s_{j,j'} \neq \perp$ for all honest j, $j' \in R'$.*

Proof. The honest node $j \in R$ computes $s_{j,j'}$ based on the set $T_{j'}$ proposed by the honest node $j' \in R$. When the node j' chooses its $T_{j'}$ and l, it checks that (for all $k \in T_{j'}$) VSS_k has outputted with success bit $b = 1$ the share $(\hat{\boldsymbol{s}}_{j',k}, \hat{\boldsymbol{w}}_{j',k}, \hat{\boldsymbol{v}}_k)$ the l-th entry non-empty $(\hat{s}_{i,k,l}, \hat{w}_{i,k,l}, \hat{v}_{k,l}) \neq \perp$. Due to guaranteed output of VSS_k, the honest node j must receive a share $(\hat{s}_{j,k,l}, \hat{w}_{j,k,l}, \hat{v}_{k,l}) \neq \perp$. Moreover, the node j' must have received a verified share-proof$_k$ with grade $b = 1$ from GC_k. Due to the graded consistency of GC, the share-proof$_k$ is also received and verified by node j. Therefore, if the node j' can compute $T_{j'}$ and l, then node j can compute the share $(s_{j,j'}, w_{j,j'}, v_{j'}) \neq \perp$.

The rest of the proof shows that the honest node j' can always compute $T_{j'}$ and l. Let $T \subset R$ be a set of $f+1$ honest nodes. Each node $i \in T$ honestly shares $s_{i,k} \neq \perp$ via VSS_i, which makes j' receive $(\hat{s}_{j',i,k}, \hat{w}_{j',i,k}, \hat{v}_{i,k}) \neq \perp$ with success bit $b = 1$ (due to the validity property of VSS). Node i also honestly sends a verified share-proof$_i$ via GC_i, making node j' receive share-proof$_i$ with grade $b = 1$ (due to the validity property of GC) and verify it. Therefore, node j' can choose $T_{j'} = T$ and $l = k$. □