Blockchain-Anchored Proactive Refresh for Dynamic Threshold Services

Seunghyun Cho[1], Jaeyeon Lee[1], Myeongjun Kim[2], Hyunseok Jeong[2], and Young-Sik Kim[1,2]

[1] Department of Electrical Engineering and Computer Science, DGIST
seunghyuncho@dgist.ac.kr, ljy8733@dgist.ac.kr, ysk@dgist.ac.kr
[2] The Interdisciplinary Studies of Artificial Intelligence, DGIST
sanmaru98u@dgist.ac.kr, hsj@dgist.ac.kr

Abstract. A long-lived threshold service must issue shares to participants that join after deployment, refresh them against a mobile adversary, and allow participants offline during a refresh to recover missed updates. We show that charging exposure solely to committed epochs is fundamentally insufficient: an aborted refresh attempt that publishes candidate update tokens can reveal the secret even if no committed epoch exceeds its corruption budget. To eliminate this exposure leakage, we index security by fine-grained states covering both committed epochs and aborted attempts. Each state's budget bounds all participants whose local material or public update records originate from that state, enforced via an append-only, durable release ledger that is never rolled back upon abort. A fixed dealer committee anchors encrypted update records in blockchain blob storage, enabling non-interactive recovery for offline participants without contacting dealers, while an interactive bisection dispute protocol deterministically attributes faulty records on chain. Our core cryptographic result is an explicit distribution-preserving shift of the actual generation transcript, establishing proactive secrecy and past-share unforgeability after dealer-state disclosure without idealizing the underlying subprotocols. An end-to-end Rust implementation benchmarks complete lifecycles across up to 1,280 participants on an EVM chain, demonstrating sub-second group recovery and practical gas costs.

Keywords: Proactive secret sharing · Verifiable secret sharing · Threshold cryptography · Blockchain data availability · Accountability.

1 Introduction

Secret sharing [47,7] underlies distributed key generation and threshold signing [25,24]. A long-running service must also admit participants after deployment, keep issued shares publicly verifiable, withstand corruptions that move between parties, and serve participants that are unreachable during a refresh. Verifiable secret sharing [14,20,42], proactive secret sharing [32] and evolving secret sharing [36,22] address these goals separately; we study the exposure accounting that is needed when they share one lifecycle.

Setting. A committee of operators, the *dealers*, maintains a long-lived secret s in shared form, and a growing population of *participants* holds issued shares so that any t_e of them can recover s without the committee, for instance if it ceases to operate. Since any k_D dealers also hold a sharing of s, the participant threshold adds an access path rather than restricting the dealers, and secrecy requires that fewer than k_D dealers and fewer than t_σ participant shares be exposed in every state σ. Participants are intermittently online and may be compromised over time. The dealers run issuance, refresh and re-issuance, whereas the t_e-of-N participant path serves only reconstruction without them. Public recovery requires a participant-corruption budget $\delta \leq t - 2$ (Section 2); a service that must tolerate $t - 1$ corrupted participants uses re-issuance instead. The dealer committee is fixed. As an intended use, consider operators, trusted up to f_D per state as in $(n_D, k_D, f_D) = (7, 3, 2)$, who maintain a treasury key for 2,048 stakeholders with $t = 1024$, so that any half can recover it should the operators become permanently unavailable; with $\delta = 32$, up to 991 stakeholders in one charged set can miss a refresh and recover from the public log within the blob retention window of about 18 days (Section 6), while participants returning past this window are serviced via re-issuance.

Exposure leakage from aborted refreshes. Consider an epoch with participant threshold $t_e = 3$ and sharing polynomial $P^{(e)}$, and suppose the adversary has corrupted participants 1 and 2, so the epoch is within budget. The committee attempts a transition to threshold 2 with candidate $\widehat{P}(x) = s + ax$ and sends each participant i the update token $\delta_i = \widehat{P}(i) - P^{(e)}(i)$. The two corrupted participants add their tokens to their source shares, obtain $\widehat{P}(1)$ and $\widehat{P}(2)$, compute $s = 2\widehat{P}(1) - \widehat{P}(2)$, and withhold their acknowledgments so that the attempt is abandoned. No committed epoch exceeded its budget, yet the secret is revealed. Imposing independent caps on each attempt is likewise insufficient: with source threshold $t_\sigma = 4$ and one corrupted participant, three aborted attempts that each publish to a different single recipient each satisfy a cap that counts only the corrupted participant and that attempt's recipients, while the union of all recipients served from the source state, together with the corrupted participant, reaches four. Our model bounds that union. Exceeding the bound does not by itself reveal s, so the condition is sufficient, not tight (Proposition 1 in Appendix B.3).

Contributions. With later issuance and public offline recovery, we account per state for the residual exposure produced by an aborted refresh, enforce a conservative sufficient condition by a durable reservation before release, and prove it over the actual transcript. (i) *Release-accounted model.* Budgets are indexed by every state, including the candidates of aborted attempts, and charged to material rather than to time; a ledger extended before any token is computed and never rolled back enforces them (Lemma 1). (ii) *Shift invariance.* Candidate states come from the same transcript as committed ones, so a shift must agree with published differences at both endpoints, complaint answers and unselected resharing contributions, on sets fixed only adaptively; we construct one

(Lemma 3), which yields proactive secrecy (Theorem 1) and past-share unforgeability (Theorem 2). (iii) *Recovery with attribution.* Offline participants recover from signed, encrypted, anchored records without contacting the committee (Lemma 2); we use the encryption of [30] unchanged and show that our log meets its query restriction (Lemma 4, Theorem 3), while proactive secrecy does not depend on it (Remark 1).

Related work. Kondi et al. [37] refresh threshold wallets with offline devices and prove an impossibility for dishonest majorities. We separate the parties that run the refresh (dealers, which hold coefficient shares) from the holders of issued shares (participants), as Lambić [39] does for a fixed signing set, and additionally support issuance to a growing set. Non-interactive resharing [29] and publicly verifiable secret sharing [27] verify every delivered share before use, whereas our delivery path invokes public verification only upon dispute. Dynamic-committee proactive secret sharing [41,50,54] transfers the shared secret across successive committees. Later issuance is the enrollment problem of repairable threshold schemes [38], which there is solved by share holders rather than by a separate committee, and accountability in [8] identifies signing quorums rather than faulty records.

Relation to existing accounting. Proactive models charge a party corrupted during an update phase to both adjacent periods [32], and dynamic-committee models do so for handoffs [41,54]; these rules account for *who* is corrupted while a refresh runs. We must also account for the *persistent residual state* left by a refresh: candidate tokens that persist in a public log after an abort, archived records exposed upon subsequent key compromise, and single dealer partials held by enrolling participants. State-indexed, material-based budgets capture these, sufficiently but not necessarily (Proposition 1); Appendix F compares the observable executions of the closest models, without implying that those schemes are insecure within their original threat models.

2 Model and Release Accounting

Parties, state, and corruption. Let λ be the security parameter; all adversaries are probabilistic polynomial time, and all failure probabilities are negligible in λ. The system consists of a data owner, a fixed committee $\mathcal{D} = \{D_1, \ldots, D_{n_D}\}$ with threshold k_D and fixed coordinates $z_j \in \mathbb{Z}_q \backslash \{0\}$, a growing participant set $\mathcal{P}$, and a blockchain $\mathbb{B}$. The owner initializes (s, r_0), with r_0 uniform and independent of s, distributes the pair in shared form, and erases its copies. Epoch e has threshold t_e, a sharing polynomial $P^{(e)}$ of degree less than t_e with $P^{(e)}(0) = s$, and an independent masking polynomial $R^{(e)}$ with $R^{(e)}(0) = r_0$; its public state consists of e, t_e and the coefficient-commitment vector $\mathbf{C}^{(e)}$. Dealer D_j retains only its coefficient shares, participant i holds $\mathsf{sh}_i^{(e)} = (P^{(e)}(i), R^{(e)}(i))$, and thresholds may change in either direction. The dealer layer runs in synchronous rounds, a property of the committee's own network that is independent of participant availability; Section 4.1 shows that this is a safety assumption. The blockchain records

finalized epoch states, registry roots and anchors for public update records, which are held in the data blobs of [9], retained by the network only for a bounded window.

A single mobile adversary observes the public transcript and may adaptively invoke CorruptDealer, which reveals the entire state of a dealer at that time; CorruptParticipant, which reveals a participant's share, long-term decryption key, staged token and temporary values; and CorruptKey(i), which reveals sk_i alone. The adversary retains all acquired information. A dealer erases its contribution and subsharing polynomials and the subshares it received on completion of each generation subprotocol, and its source coefficient shares at commit; a participant erases its source share, staged token and plaintext partials once it has observed commit and applied the update. Private deliveries run over forward-secret channels whose session state is erased on completion.

Charging criterion. A time-indexed interpretation of $\mathcal{P}_\sigma^{\mathsf{cor}}$ is inadequate: a participant that stages a token and crashes before applying it still holds its source share after the committee has erased the source dealer state, and single dealer partials $A_\sigma(z_j, i)$ held by four participants can determine s although none of them forms a share (Appendix B.2). We therefore charge material:

> $i \in \mathcal{P}_\sigma^{\mathsf{cor}}$ if the corruption of i, *whenever it occurs*, returns local state that exposes the σ-share of i, any private partial evaluation of the σ-state at i, or staged update material in which σ is the source or the target.

CorruptKey is covered by the public-record charge of (1). The bound $|\mathcal{P}_\sigma^{\mathsf{cor}}| \le \delta$ is an assumption about the adversary's access to material, which the release ledger does not enforce.

States, exposure, and the release ledger. A transition attempt produces a *candidate state*: a fresh pair of epoch polynomials together with the tokens that move participants onto it. We index the model by *states*, namely the epoch-0 state and the candidate state of every attempt, and write $\mathcal{D}_\sigma^{\mathsf{cor}}$ and $\mathcal{P}_\sigma^{\mathsf{cor}}$ for the dealers and participants exposed at σ, $\mathcal{O}_\sigma$ for the participants moved onto σ through the public record, and $\mathcal{O}_a$ for the public recipients of an attempt a. Put

$$U_\sigma^{\mathsf{rel}} := \bigcup_{\substack{a:\ \mathrm{src}(a)=\sigma \\ a\ \text{released}}} \mathcal{O}_a, \quad \widehat{U}_\sigma := \bigcup_{\substack{a:\ \mathrm{src}(a)=\sigma \\ a\ \text{reserved}}} \mathcal{O}_a, \quad \mathcal{E}_{P,\sigma} := \mathcal{P}_\sigma^{\mathsf{cor}} \cup \mathcal{O}_\sigma \cup U_\sigma^{\mathsf{rel}}. \tag{1}$$

An adversary that holds sk_i and the σ-share of $i \in U_\sigma^{\mathsf{rel}}$ derives the candidate share from public data, which is why the released set is charged; the policy is enforced against the larger $\widehat{U}_\sigma$, since a dealer that crashes between reserving and publishing cannot determine whether the release completed. Both unions run over *all* attempts, aborted ones included, so $\widehat{U}_\sigma$ is a durable *release ledger*.

Definition 1 (Admissible mobile adversary). *A probabilistic polynomial-time mobile adversary is* admissible *if every state σ of the execution, committed or aborted, satisfies $|\mathcal{D}_\sigma^{\mathsf{cor}}| \le f_D < k_D$ and $|\mathcal{E}_{P,\sigma}| < t_\sigma$.*

For an attempt a from σ with target threshold t_a, public-recipient set $\mathcal{O}_a$ and eligible participant set $\mathcal{P}_a$, the committee evaluates

$$\delta + \left|\mathcal{O}_\sigma \cup \widehat{U}_\sigma \cup \mathcal{O}_a\right| < t_\sigma, \qquad \delta + |\mathcal{O}_a| + \rho_a^{\mathsf{out}} < t_a, \qquad |\mathcal{P}_a| \geq t_a, \tag{2}$$

where δ is the declared participant-corruption budget and ρ_a^{out} is a planning reservation that the target state sets aside for its own future public recipients; Lemma 1 does not depend on it. With $\rho_a^{\mathsf{out}} = 0$ the second condition reads $|\mathcal{O}_a| \leq t_a - \delta - 1$, so a public path exists only if $\delta \leq t_a - 2$. Because a candidate state that exists is a state of the model even if no token is ever computed from it, an attempt proceeds in five strictly sequential steps, none of which commences before the preceding step completes. (i) *Target pre-check*: the second and third conditions of (2) are verified; if either fails, no state is created. (ii) *Candidate generation*: σ_a is generated by (6) with a fresh nonce ν, and each dealer signs its target commitment vector. (iii) *Reservation*: the first condition is evaluated against the current ledger through $\mathsf{ReserveRelease}(\sigma, \mathsf{root}_{\mathsf{old}}, \nu, \mathcal{O}_a)$, a compare-and-set operation linearized by a totally ordered dealer-side log and certified by $n_D - f_D$ signatures (Appendix B.4); on failure, the candidate is erased and the ledger is unchanged. (iv) *Release*: an honest dealer computes or sends a token only after the certificate exists; reclassifying an online participant as a public recipient repeats steps (i) and (iii), extending the ledger first. (v) *Finalization*: the attempt commits or aborts; an abort erases the candidate state but not the ledger.

Lemma 1 (The release rule implies admissibility). *Suppose every transition attempt follows the procedure above and $|\mathcal{P}_\sigma^{\mathsf{cor}}| \leq \delta$ for every state. Then $|\mathcal{E}_{P,\sigma}| < t_\sigma$ for every state of the execution: committed, aborted after release, or abandoned at reservation.*

Proof (sketch). A released attempt is a reserved one, so $U_\sigma^{\mathsf{rel}} \subseteq \widehat{U}_\sigma^{\mathsf{fin}}$ and (1) gives $|\mathcal{E}_{P,\sigma}| \leq \delta + |\mathcal{O}_\sigma \cup \widehat{U}_\sigma^{\mathsf{fin}}| < t_\sigma$, the last step being the source condition of the last attempt that reserved or, when none did, the target pre-check of the attempt that created σ. Appendix B.1 gives the case analysis. □

For the variant that keeps only the per-attempt cap, Proposition 1 (Appendix B.3) exhibits an execution that violates Definition 1 although the adversarial view is consistent with every value of the secret; the rule is therefore sufficient but not necessary.

Security goals and realization boundary. Correctness requires that combination and update return the epoch share and mask, that verification accepts both, and that any t_e valid shares reconstruct s. *Lifecycle integrity* requires that violations of one-time issuance, of consistency among accepted shares, and of canonical finalization have negligible probability against f_D deviating dealers. *Proactive secrecy* is the indistinguishability of two secrets chosen by an admissible adversary that drives polynomially many epochs, and *forward security* (Section 4.2) protects unexposed past shares against forgery after the complete dealer state

is disclosed; Appendix A.2 states the experiments. We prove *conditional security* for executions that respect Definition 1. *Realization*, that a deployed service produces only such executions, additionally assumes forward-secret private delivery, synchrony of the dealer layer, a linearizable release ledger with a separate dealer reservation key (Appendix B.4), authentication-key recovery anchored outside the committee, and retrievability of public records and dispute inputs within the blob retention window, currently 4,096 epochs, or about 18 days, on Ethereum [16].

3 The Protocol

Building blocks. $\mathbb{G}$ has prime order q with generators g and a transparently derived h whose discrete logarithm $\alpha = \log_g h$ is known to no party. Shamir sharing [47] is used at two layers: $[x]$ denotes a sharing of x by a polynomial of degree less than k_D of which D_j holds the evaluation at z_j, the epoch sharing polynomial being the second layer, and $\lambda_j^S(x)$ and $\lambda_i^T(x)$ are the Lagrange coefficients over dealer coordinates and over participant identifiers. A Pedersen verifiable sharing of (m, r) with threshold τ publishes the coefficient commitments $g^{a_\ell} h^{r_\ell}$, $\ell < \tau$, of a sharing polynomial with constant term m and an independent masking polynomial with constant term r; the commitment is perfectly hiding, computationally binding and additively homomorphic [42]. We write ϵ_{VSS}, ϵ_{bc}, ϵ_{bind} and ϵ_{sig} for the failure probabilities of one inconsistent sharing accepted by all honest parties, of one reliable-broadcast instance, of binding and of signature unforgeability, and Q_{gen} and Q_{bc} for the numbers of sharing and broadcast instances.

Assumption 1 (Dealer parameters). (P1) $k_D \geq f_D + 1$ *and* (P2) $n_D \geq 2f_D + k_D$, *where* f_D *bounds* $|\mathcal{D}_\sigma^{\mathsf{cor}}|$ *at every state and the number of dealers that deviate within one generation invocation. A mobile adversary may corrupt more than* f_D *dealers over a long execution; the bound applies to the set charged to one state.*

(P1) gives secrecy against f_D corrupted dealers; (P2) lets the honest dealers alone supply k_D correct contributions and places at least $n_D - 2f_D \geq k_D$ honest members in any published family of per-dealer commitments. Two linear subprotocols, neither of which multiplies shared values, generate all dealer-side state (Appendix D.1). $\mathsf{ReShare}([x])$ replaces a sharing of x by a new sharing of the same value: each dealer reshares its own share under a Pedersen verifiable sharing whose constant-term commitment is publicly derived from the source commitments, and the new share is the Lagrange combination over a broadcast-agreed set S of the k_D smallest-indexed accepted resharers. $\mathsf{JRSS}(\nu, \ell)$ adds independently sampled sharings of the qualified dealers [43,25], with a per-attempt nonce ν in the domain separator.

Each dealer holds a signing key pair, and each participant holds a key pair for the multi-recipient encryption scheme of [30], which we adopt without modification (Appendix D.2). We rely on three of its properties: every ciphertext

carries a proof of possession π^{epk} of its ephemeral exponent, which decryption checks; a recipient can prove what a ciphertext decrypts to [13]; and the scheme is associated-data-only chosen-ciphertext secure, with advantage $\mathsf{Adv}^{\mathsf{ado}}$, admitting decryption and decryption-proof queries on every ciphertext whose associated data was never submitted to the encryption oracle. The scheme is not an authenticated encryption scheme [3,44,23], so record integrity rests on the dealer signature and the anchor.

Epoch state, issuance, and verification. The epoch polynomials are $P^{(e)}(x) = \sum_{\ell<t_e} a_\ell^{(e)} x^\ell$ with $a_0^{(e)} = s$ and $R^{(e)}$ likewise with $r_0^{(e)} = r_0$, and D_j holds $(a_{j,\ell}^{(e)}, r_{j,\ell}^{(e)})$, the evaluations at z_j of the degree-(k_D-1) sharings of those coefficients; $\mathsf{ShareInit}$ shares (s, r_0) once with threshold k_D and draws the remaining coefficients from $\mathsf{JRSS}(\nu_0, \ell)$. For a vector $\mathbf{X}$ of group elements we write $\mathbf{X}(y) := \prod_\ell X_\ell^{y^\ell}$. Dealer D_j signs and publishes $C_{j,\ell}^{(e)} = g^{a_{j,\ell}^{(e)}} h^{r_{j,\ell}^{(e)}}$. Let $\mathcal{Q}_D^{(e)}$ be the dealers whose signed vector equals the per-dealer commitment vector derived from the broadcast generation transcript, so that $|\mathcal{Q}_D^{(e)}| \geq n_D - f_D$; a differing vector is excluded rather than blocking the record. The record accepts, against a canonical $S_0 \subseteq \mathcal{Q}_D^{(e)}$ of size k_D, only valid signatures and

$$C_{j,\ell}^{(e)} = \prod_{u \in S_0} \big(C_{u,\ell}^{(e)}\big)^{\lambda_u^{S_0}(z_j)}, \qquad D_j \in \mathcal{Q}_D^{(e)}, \tag{3}$$

which fixes the degree-(k_D-1) commitment polynomial of every coefficient, since at least k_D accepted vectors are honest. Any S with $|S| \geq k_D$ then yields

$$C_\ell^{(e)} = \prod_{j \in S} \big(C_{j,\ell}^{(e)}\big)^{\lambda_j^S(0)} = g^{a_\ell^{(e)}} h^{r_\ell^{(e)}}, \tag{4}$$

and $\mathbf{C}^{(e)}$, published once per epoch, is the public verification state; enrollments do not modify it.

An enrollment request reserves the identifier and public key before any dealer responds. Dealer D_j encrypts the pair $(v_{j,i}^{(e)}, \rho_{j,i}^{(e)})$ of evaluations of its coefficient shares at i under pk_i and signs an attestation containing $\mathbf{C}_j^{(e)}(i)$, so the completion certificate depends on public state and signatures alone (Lemma 6). The participant batch-verifies the k_D partials by a random linear combination [4] whose Fiat–Shamir challenges [21] hash every decrypted partial with a random oracle $H_{\mathsf{batch}} : \{0,1\}^* \to \mathbb{Z}_q$, instantiated by hash-to-field [18]. The test rejects, except with probability $(q_H+1)/q$ for q_H adversarial queries, whenever some *residual* $g^{v_{j,i}^{(e)}} h^{\rho_{j,i}^{(e)}} \mathbf{C}_j^{(e)}(i)^{-1}$ is nontrivial (Lemma 5, Appendix D.5). It then obtains $v_i^{(e)} = \sum_{j \in S} \lambda_j^S(0) v_{j,i}^{(e)} = P^{(e)}(i)$ and likewise $\rho_i^{(e)} = R^{(e)}(i)$, and a share is accepted exactly when

$$g^{v_i^{(e)}} h^{\rho_i^{(e)}} \stackrel{?}{=} \mathbf{C}^{(e)}(i). \tag{5}$$

A registry enforces one-time issuance and re-issues the *current* share to a participant that holds fewer than k_D valid update records (Appendix D.4).

Transition, tokens, and finalization. Step (ii) of the attempt procedure computes

$$\big([a_0^{(e+1)}],[r_0^{(e+1)}]\big) \leftarrow \mathsf{ReShare}\big([a_0^{(e)}],[r_0^{(e)}]\big), \quad \big([a_\ell^{(e+1)}],[r_\ell^{(e+1)}]\big) \leftarrow \mathsf{JRSS}(\nu,\ell) \tag{6}$$

for $1 \le \ell < t_{e+1}$, and each dealer signs its target vector, which must satisfy (3); a transition thus consists of $n_D t_{e+1}$ Pedersen verifiable sharings of degree k_D-1 and local linear combinations. Let $T = \max(t_e, t_{e+1})$, with source and target coefficients *both* zero-padded to length T and missing commitments set to $1_{\mathbb{G}}$, which matters when the threshold decreases. Dealer D_j signs $\Delta_{j,\ell} = g^{\Delta a_{j,\ell}^{(e)}} h^{\Delta r_{j,\ell}^{(e)}}$ from the differences of its padded coefficient shares, so $\Delta_0 = 1_{\mathbb{G}}$ and $\Delta_\ell = C_\ell^{(e+1)}(C_\ell^{(e)})^{-1}$. For recipient i the dealer stages $\mathsf{pupd}_{j,i}$, the evaluations of its own delta vector at i; combining k_D valid partials gives $\delta v_i = P^{(e+1)}(i) - P^{(e)}(i)$ and $\delta\rho_i = R^{(e+1)}(i) - R^{(e)}(i)$, accepted when

$$g^{\delta v_i} h^{\delta\rho_i} \stackrel{?}{=} \boldsymbol{\Delta}(i), \tag{7}$$

whether the threshold increases, decreases, or remains unchanged. With $\mathsf{Pending}_e$ the pending epoch-e activations, $\mathcal{R}_e$ the participants to serve, q_i^{sync} the valid direct partials that i acknowledged after durable staging, and q_i^{async} the distinct signed records anchored for i, the attempt commits exactly when the target commitment record is accepted and

$$\begin{aligned} \mathsf{Pending}_e = \varnothing, \qquad & |\mathcal{Q}_D^{(e+1)}| \ge n_D - f_D, \\ q_i^{\mathsf{sync}} \ge k_D \ (i \in \mathcal{R}_e \setminus \mathcal{O}_{e+1}), \qquad & q_i^{\mathsf{async}} \ge n_D - f_D \ (i \in \mathcal{R}_e \cap \mathcal{O}_{e+1}). \end{aligned} \tag{8}$$

Commit applies each staged direct token once and erases the source coefficient shares, the temporary sharing state and every plaintext partial. Since at most f_D published records are malformed, the publication quorum leaves

$$(n_D - f_D) - f_D = k_D + m, \qquad m := n_D - 2f_D - k_D \ \ge 0 \tag{9}$$

correct records, so k_D valid records survive the loss of up to m of them.

Delivery and offline recovery. For an online recipient, D_j sends $\mathsf{pupd}_{j,i}$ over the confidential channel; the participant combines k_D valid partials, checks (7) and stages the token until commit or abort. For $i \in \mathcal{O}_{e+1}$, D_j computes the *evaluation commitment* $E_{j,i} = g^{\delta v_{j,i}} h^{\delta\rho_{j,i}}$, fixes $\mathsf{ad}_{j,i} = (e, e{+}1, \nu, \mathsf{mode}, j, i,\ H(\boldsymbol{\Delta}_j), E_{j,i})$, where mode is the record format of the attempt, and publishes the signed record

$$\mathsf{rec}_{j,i} = \big(\mathsf{ad}_{j,i}, \chi_{j,i}, c_{j,i}, \sigma_{j,i}\big), \quad (\chi_{j,i}; c_{j,i}) \leftarrow \mathsf{E}\big(\mathsf{ad}_{j,i}; (i, \mathsf{pk}_i, \mathsf{pupd}_{j,i})\big), \tag{10}$$

after which it erases its ephemeral randomness. Each record is a separate invocation of E under its own associated data, which keeps the reduction of Theorem 3 simple at the cost of the amortization that a per-dealer multi-recipient invocation would give. A record enters the anchored log only if its signature verifies under the record key currently registered for dealer j, and duplicates of (e, ν, j, i)

are rejected; the log digest is checked against the native blob anchor [9]. A returning participant accepts a record if its signature, anchor membership proof and proof of possession verify, and retains it if the decrypted plaintext opens $E_{j,i}$. *Aggregate phase:* it combines the canonical k_D kept records and tests (7); on acceptance, it applies the token. *Localization phase:* otherwise, it tests for each candidate

$$E_{j,i} \stackrel{?}{=} \boldsymbol{\Delta}_j(i) = \prod_{\ell<T} \Delta_{j,\ell}^{i^\ell}, \qquad \Delta_{j,\ell} = C_{j,\ell}^{(e+1)} \big(C_{j,\ell}^{(e)}\big)^{-1}, \tag{11}$$

recomputed from the two anchored commitment vectors, discards failing records and recombines. Neither phase contacts a dealer.

Lemma 2 (Robust recovery). *Call a record* valid *for i if it passes admission, its plaintext opens $E_{j,i}$, and it satisfies* (11)*. If i holds its source share and can retrieve, within the retention window, all but at most m of the valid records fixed by finalization, together with the authenticated epoch states and the anchored per-dealer commitment vectors of both epochs, then, except with the failure probabilities of the signature, anchor and binding assumptions, the algorithm returns the epoch-$(e{+}1)$ share of i after reading at most $k_D + f_D$ records. If it terminates in the aggregate phase, the token is correct but the combined records are not certified individually, since errors in several records can cancel; if it terminates after localization, all k_D combined records are valid.*

The proof is in Appendix D.6; an implementation that terminates in the aggregate phase must therefore not record the contributing dealers as having produced valid records. Blob data is retained for about 18 days [16], and nodes do not store complete blobs under data-availability sampling [19,40], so participants that return later are recovered by re-issuance.

4 Security Analysis

4.1 Shift Invariance of the Transcript

We write the dealer-shared coefficient state of σ as the bivariate polynomials

$$A_\sigma(z,x) = \sum_{\ell<t_\sigma} A_{\sigma,\ell}(z)\, x^\ell \quad \text{and} \quad M_\sigma(z,x),$$

of degree less than k_D in z, so that $A_\sigma(0,x) = P^{(\sigma)}(x)$ and $M_\sigma(0,x) = R^{(\sigma)}(x)$. Dealer D_j holds the row at z_j, so a dealer corruption exposes a row and a participant exposure a column. The argument uses $\alpha = \log_g h$ only to analyze distributions. For finite $Z \subset \mathbb{Z}_q \setminus \{0\}$ put

$$\Pi_Z(y) := \prod_{u\in Z} \frac{y-u}{-u}, \qquad \text{so} \qquad \Pi_Z(0) = 1, \quad \Pi_Z|_Z = 0, \quad \deg \Pi_Z = |Z|, \tag{12}$$

with $\Pi_\varnothing \equiv 1$.

Lemma 3 (Shift invariance). *Consider an execution against an admissible adversary in which honest parties erase as specified, private deliveries are ideal confidential channels, every sharing accepted by all honest dealers is consistent of degree k_D-1 with an agreed qualified set, and every batch test accepts exactly when all of its residuals are trivial. For every $\Delta \in \mathbb{Z}_q$ there is a bijection on the honest parties' randomness that leaves unchanged every published commitment, every message the adversary receives, and every public update record, and that replaces s by $s+\Delta$. The dealer rows are shifted, so the lemma does not apply to an experiment that discloses a complete dealer state.*

Proof. Exposure sets. For a transition from σ to σ' by ReShare over an accepted S of size k_D, a dealer corrupted during the transition reveals the subshares it received, and a contributor d that answers a complaint broadcasts the disputed pair. With B the contributors corrupted during the transition and R_d the coordinates at which a subsharing of an honest contributor d, selected or not, in ReShare or JRSS of this transition was opened, the construction must respect

$$C_{\mathsf{tr}} := \big(\mathcal{D}^{\mathsf{cor}}_{\sigma} \cap \mathcal{D}^{\mathsf{cor}}_{\sigma'}\big) \cup \bigcup_{d \notin B} R_d, \qquad \big|C_{\mathsf{tr}} \cup \mathcal{D}^{\mathsf{cor}}_{\sigma'}\big| \le k_D - 1. \tag{13}$$

Under synchrony only a corrupted recipient can complain against an honest contributor, so $R_d \subseteq \mathcal{D}^{\mathsf{cor}}_{\sigma} \cap \mathcal{D}^{\mathsf{cor}}_{\sigma'}$ for $d \notin B$ and $|C_{\mathsf{tr}} \cup \mathcal{D}^{\mathsf{cor}}_{\sigma'}| \le f_D \le k_D - 1$. Synchrony is therefore a safety assumption: with $k_D = f_D + 1$, as in every committee of Section 6, one complaint answer at an honest coordinate while f_D dealers are corrupted can determine s, and aborting cannot revoke it (Appendix C.5). Without synchrony, (13) must be enforced directly: a contributor answers a complaint only if at most $k_D - 1 - f_D$ coordinates, including the new one, have been opened in the transition, and otherwise stops; raising k_D merely provides the necessary threshold slack. Alternatively, publicly verifiable dealing [29,27] avoids opening subshares. With $C^{\mathsf{in}}_{\mathsf{tr}}(\sigma)$ the set (13) of the transition that created σ, empty for σ_0, put $Z_\sigma := \mathcal{D}^{\mathsf{cor}}_{\sigma} \cup C^{\mathsf{in}}_{\mathsf{tr}}(\sigma)$.

Construction. For every state define

$$j_\sigma := \Delta \cdot \Pi_{Z_\sigma}, \qquad \deg j_\sigma = |Z_\sigma| < k_D, \quad j_\sigma(0) = \Delta, \quad j_\sigma|_{Z_\sigma} = 0. \tag{14}$$

At σ_0 the owner's sharing polynomial of s becomes $A_{\sigma_0,0} + j_{\sigma_0}$ and its masking polynomial $M_{\sigma_0,0} - \alpha^{-1} j_{\sigma_0}$. Across a transition every dealer, not only those later selected, has reshared its source share, so the shift corrects *every* executed honest contribution on the *common* set C_{tr} before retargeting the aggregate. Writing $\widetilde{j}_{\sigma'} := \sum_{j \in S} \lambda^S_j(0) W^{\mathsf{base}}_j$, put

$$W^{\mathsf{base}}_d := j_\sigma(z_d) \cdot \Pi_{C_{\mathsf{tr}}}, \qquad \widetilde{j}_{\sigma'} = \Delta \cdot \Pi_{C_{\mathsf{tr}}}, \qquad j_{\sigma'} = \Delta \cdot \Pi_{Z_{\sigma'}}, \tag{15}$$

the middle identity by exactness of Lagrange interpolation. Fix the smallest $h \in S \setminus B$, which exists because $|S \setminus B| \ge k_D - f_D \ge 1$, and set

$$W_h := W^{\mathsf{base}}_h + \frac{j_{\sigma'} - \widetilde{j}_{\sigma'}}{\lambda^S_h(0)}, \qquad W_j := W^{\mathsf{base}}_j \quad (j \ne h), \tag{16}$$

replacing the subsharing polynomial V_j of D_j by $V_j + W_j$ and its masking counterpart by $V'_j - \alpha^{-1}W_j$. For the higher coefficients put

$$J_\sigma(z,x) := j_\sigma(z) \cdot \Pi_{\mathcal{E}_{P,\sigma}}(x), \qquad A_\sigma \mapsto A_\sigma + J_\sigma, \qquad M_\sigma \mapsto M_\sigma - \alpha^{-1}J_\sigma, \tag{17}$$

realized by adding the coefficients of index $\ell \geq 1$ of J_σ to the contribution of the smallest JRSS contributor h' outside $\mathcal{D}^{\mathsf{cor}}_\sigma$, which exists by (P2), and subtracting α^{-1} times them from its masking contribution.

Verification. Degrees: $\deg W_j \leq k_D - 1$ by (13), and $\deg_x J_\sigma = |\mathcal{E}_{P,\sigma}| < t_\sigma$ by Definition 1, which is where admissibility enters. Honest programs: the correction of (16) vanishes at 0, so every resharing still opens the public commitment to its shifted source share. Opened subshares: every W_j vanishes on C_{tr}, so every value revealed by a corruption or a complaint answer is unchanged. Aggregate: $\sum_{j\in S} \lambda^S_j(0) W_j = j_{\sigma'}$. Commitments: $g^{A+J}h^{M-\alpha^{-1}J} = g^A h^M$ at every publicly fixed point. Rows, columns and tokens: J_σ vanishes on every corrupted row and on the whole column of every $i \in \mathcal{E}_{P,\sigma}$, so every share, single dealer partial or staged material returned by a corruption is preserved, and a recipient moved from σ to σ' lies in $\mathcal{E}_{P,\sigma} \cap \mathcal{E}_{P,\sigma'}$, so published differences, evaluation commitments and ciphertexts are unchanged. Adversarial contributions are never modified.

Bijection. The corrections depend on sets fixed only at the end of the execution; inducting on adversary-observable events shows that the shifted execution yields the same view and the same sets, because every honest decision depends only on verification outcomes and public data. For fixed sets each correction is linear in Δ, so $-\Delta$ inverts the map, and each replaced coordinate is free and uniform. Appendix C.2 gives the full induction. □

4.2 Proactive Secrecy and Forward Security

Let $\epsilon_{\mathsf{sync}}(Q_{\mathsf{sync}})$ be the advantage of distinguishing the Q_{sync} private deliveries of the execution (enrollment partials, direct tokens, re-issuance and dealer-to-dealer channels) from ideal confidential channels (Appendix A.1), let q_B be the number of batch tests, including those of recursive splitting, and let q_H be the number of the adversary's queries to H_{batch}.

Theorem 1 (Proactive secrecy). *For every admissible mobile adversary $\mathcal{A}$, in the random-oracle model for the batch challenges,* $\mathsf{Adv}^{\mathsf{ps}}_{\Pi,\mathcal{A}} \leq \epsilon_{\mathsf{sync}}(Q_{\mathsf{sync}}) + Q_{\mathsf{gen}}\epsilon_{\mathsf{VSS}} + Q_{\mathsf{bc}}\epsilon_{\mathsf{bc}} + q_B(q_H+1)/q$.

Proof. Game 1 replaces every private delivery by an ideal channel, at cost $\epsilon_{\mathsf{sync}}(Q_{\mathsf{sync}})$. Game 2 aborts when an accepted sharing is inconsistent or a broadcast instance fails, at cost $Q_{\mathsf{gen}}\epsilon_{\mathsf{VSS}} + Q_{\mathsf{bc}}\epsilon_{\mathsf{bc}}$; an accepted inconsistent subsharing would prevent the resulting shares from lying on any polynomial of degree less than k_D, rendering $A_\sigma(z,x)$ undefined. Game 3 replaces every batch test by the ideal test that accepts exactly when all residuals are trivial, at cost $q_B(q_H+1)/q$ by Lemma 5 (Appendix D.5); this game transition is necessary because the batch challenges hash decrypted partials, which the shift changes at unexposed recipients, whereas the residuals are fixed by the corrupted dealers' messages and

public commitments. Game 3 satisfies the hypotheses of Lemma 3, and the view is a function of the objects the lemma fixes, including the registry, which depends only on public state and signing keys (Lemma 6). Candidate states of aborted attempts are covered because admissibility, enforced by Lemma 1, gives the degree bound of (17) at every state. Applying Lemma 3 with $\Delta = s_1 - s_0$ makes the two game-3 views identical, and the triangle inequality gives the claim. □

Remark 1 (Scope of the public-path charge). Theorem 1 does not use the security of the record encryption: (1) charges every public recipient to both endpoints, so the shift fixes every public plaintext and ciphertext, and encryption does not enlarge the admissible offline workload. The charge is needed because keys are long-lived: an adversary that archives records and later obtains sk_i decrypts them, and the resulting relations can accumulate into a share (Proposition 3, Appendix C.6). Token privacy (Theorem 3) is separate; key-evolving encryption [29] could relax the charge.

Record format. The accounting is parameterized by the record format; the source-side charge comes from the difference format. A record may instead carry the target partial (a *snapshot*), selected by the authenticated mode field; Corollary 1 (Appendix C.1) shows that snapshots need to be charged only to the target state, and they let a participant that missed L transitions recover from k_D rather than Lk_D valid partials without its source share. Conversely, a subsequent compromise of sk_i reveals the target share, whereas a difference record reveals only a single linear relation; this advantage holds for an isolated transition (Proposition 2) but does not compose (Proposition 3), so long retention or forward-secure encryption favors snapshots. While difference records are not universally superior to snapshots, we adopt them in our primary implementation and exercise snapshots on the reference model of Section 6.

Forward security. If the complete dealer state is disclosed, s is lost, but a share for an earlier committed state $\sigma^\star$ at an identifier outside $\mathcal{E}_{P,\sigma^\star}$ still cannot be fabricated; actual issuance is attested by the registry's anchored completion certificate. Since s acts as one further evaluation, we require $|\mathcal{E}_{P,\sigma^\star}| \le t_{\sigma^\star} - 2$ (Definition 4).

Theorem 2 (Forward security: past-share unforgeability). *Under the hypotheses of Theorem 1 and Pedersen binding, every admissible $\mathcal{A}$ satisfies*

$$\Pr[\mathsf{ExpFwd}_{\Pi,\mathcal{A}} = 1] \le \epsilon_{\mathsf{sync}}(Q_{\mathsf{sync}}) + Q_{\mathsf{gen}}\epsilon_{\mathsf{VSS}} + Q_{\mathsf{bc}}\epsilon_{\mathsf{bc}} + q_B(q_H+1)/q + \epsilon_{\mathsf{bind}} + 1/q.$$

Proof (sketch). A forged share that differs from the generated one breaks binding. Otherwise, after the hops of Theorem 1, apply the construction of Lemma 3 at $\sigma^\star$ only, with uniform γ and

$$J(z,x) = \gamma \cdot \Pi_{Z_{\sigma^\star}}(z) \cdot \frac{x\, \Pi_{\mathcal{E}_{P,\sigma^\star}}(x)}{i^\star\, \Pi_{\mathcal{E}_{P,\sigma^\star}}(i^\star)}. \tag{18}$$

Since $J(z,0) \equiv 0$, s, r_0 and every later state, including the disclosed one, are unchanged, at the cost of a unit increase in degree ($\deg_x J \le t_{\sigma^\star} - 1$). J vanishes on $Z_{\sigma^\star}$ and on $\mathcal{E}_{P,\sigma^\star}$, which contains every public recipient incident to $\sigma^\star$, and $J(0, i^\star) = \gamma$, so the target share is uniform given the view (Appendix C.3). □

4.3 Privacy of the Anchored Tokens

Lemma 4 (Associated-data uniqueness). *Within one execution, two anchored records with the same associated data are the same record.*

Indeed, $\mathsf{ad}_{j,i}$ fixes $(e, e{+}1, \nu, j, i)$, nonces are fresh per attempt, and a record enters the log only under the registered record key of dealer j, with duplicates rejected. In the token-privacy experiment the challenge is the *whole family* of records addressed to one recipient $i^\star$ by the honest dealers, each encrypting its real token or zeros according to a hidden bit; a single record would be a weak challenge, since any k_D partial tokens determine the rest. The adversary controls f_D dealers, never corrupts $i^\star$, and may query decryption and complaint oracles on every non-challenge record; the challenger computes the subsequent actions of $i^\star$ from its real tokens regardless of the bit (Definition 5).

Theorem 3 (Token privacy). *If the record encryption is associated-data-only chosen-ciphertext secure in the sense of [30], then for every $\mathcal{A}$ there is a $\mathcal{B}$ with $\mathsf{Adv}^{\mathsf{tok}}_{\Pi,\mathcal{A}} \le N\mathsf{Adv}^{\mathsf{ado}}_{\mathcal{B}} + \epsilon_{\mathsf{sig}}$, where N bounds the number of participants; the factor N disappears when $i^\star$ is fixed in advance.*

Proof (sketch). $\mathcal{B}$ submits exactly the honest dealers' challenge records to its encryption oracle and encrypts all other records itself. A permitted query to $i^\star$ from a corrupted signer carries associated data that, by Lemma 4, was never submitted, and a record that claims an honest dealer's index but differs from the record that dealer published fails admission unless a signature is forged (Appendix C.4). □

5 Accountable Delivery and Disputes

Disputes, which incur zero on-chain overhead in an optimistic (fault-free) execution, separate four questions resolved from anchored data. *Admission*: a complaint names a record in the anchored log, signed under the named dealer's registered key; otherwise the claimant forfeits its bond. *Plaintext validity*: the contract checks the proof of possession, the claimant's decryption proof and the opening of $E_{j,i}$, one group equation independent of t_{e+1}; an invalid decryption proof is charged to the claimant, and a failed possession proof or opening to the dealer. *Evaluation consistency*, that is, (11), is detected off chain by a batched check and adjudicated by a bisection protocol of $\lceil \log_2 T \rceil$ rounds in the style of [33,49], whose final step the contract recomputes from the two anchored commitment vectors. *Availability*: a recipient that misses a record challenges the dealer, which must supply an opening against the finalized anchor

Table 1. Summary of the end-to-end lifecycle measurements: single executions of the workload with 16 returning participants (Appendix G.3, Appendix G.4).

End-to-end lifecycle, $(n_D, k_D, f_D) = (4,2,1), (7,3,2), (13,5,4)$			
Complete lifecycle (s)	715.480	1,001.582	1,756.229
Refresh to epoch 3 (s)	26.293	36.311	79.554
Group recovery of two missed updates (s)	0.606	1.016	2.652
Publication at epoch 2 (blobs / execution gas)	1 / 683,006	2 / 978,762	3 / 1,491,921

before a deadline or forfeit its stake. Appendix E shows that valid complaints succeed and honest dealers cannot be framed (Theorem 5), that challenges resolve only against the finalized anchor (Theorem 6), and that the bisection is complete, non-frameable, and distinguishes non-response from a computational fault (Theorem 7). These results attribute publication obligations; they provide cryptographic accountability rather than physical data availability, and we leave game-theoretic incentive analysis of withholding to future work.

6 Evaluation, Baselines, and Operational Limits

Setup. Component timing microbenchmarks are provided as supplementary reference data in Appendix G.1 rather than primary empirical evidence; size estimates use a cost model with 32-byte group elements at $\lambda = 128$. End-to-end lifecycles run our Rust implementation, with commitments and record encryption over BN254, as independent processes on one host with a local Ethereum development node, across epochs 0–3. Three workloads scale the participant population from 1,024 to 1,280 with a fixed threshold 128 or an increase to 160, and with one or 16 offline participants that miss two transitions; each runs once per committee, without injected faults (Appendix G.2).

Results. Every lifecycle completes issuance, growth, three transitions, offline recovery and reconstruction (Table 1). Generation, issuance and enrollment account for 85.2–88.6% of lifecycle time, largely because activations are confirmed sequentially on chain. With one offline participant, the higher threshold lengthens refreshes at epochs 2 and 3 by 10.5–21.3%. Returning participants read exactly k_D valid records per missed transition. An epoch update costs 552,003 execution and 131,072 blob gas (Table 10).

Operational limits. At $t = 1024$ and $\delta = 32$, (2) admits at most 991 recipients in one charged set, 495 each for disjoint incoming and outgoing sets, and 330 each with one more aborted set (Appendix G.1); snapshot records, charged only to the target, admit 991 in all three cases. With the modeled 379-byte record, a blob holds 315 records, and 2,048 offline participants need 20, 33 and 46 blobs, that is, one to three blocks at the current maximum of 21 blobs per block and two to four at the target of 14 [17]. The implemented BN254 encoding adds about 857 bytes per record (Table 17), or about 148 records per blob, and BN254 offers less than

Table 2. Failure-path traces of the executable reference model (Appendix G.5). The model checks specified behavior; it does not measure the implementation.

Scenario	Observed behavior
Abort and retry; $t_\sigma = 8$, $\delta = 1$, $\lvert\mathcal{O}_\sigma\rvert = 2$	after two aborted releases the ledger keeps $\{3, \ldots, 6\}$; $\{7\}$ is refused ($8 \not< 8$); a retry to $\{3, 4\}$ is admitted; an under-signed certificate is rejected
Crash between reservation and release	restart replays the recipient as reserved, none as released; the identical request is idempotent; a stale root is rejected
Wrong $E_{j,i}$; $(n_D, k_D, f_D) = (7, 3, 2)$	aggregate rejects; localization flags the dealer; 4 reads; share verifies
Two cancelling records; $(7, 3, 2)$	aggregate accepts the correct token after 3 reads; 2 of the 3 combined records violate (11), and none is certified
Forged signature, bad possession proof, failed opening; $(7, 3, 2)$	discarded before combining; 4 reads; complaints charge the dealer, or the claimant for an honest record

the 128-bit security of the cost model [51]. The epoch-update figure excludes the per-dealer commitment vectors required for dispute resolution (Appendix D.3).

Failure-path checks. The lifecycle runs inject no faults, so we exercise the failure paths on an executable reference model, separate from the Rust implementation, with signatures, record encryption, anchors and certified reservations on a durable ledger over the group of [35] (Table 2, Appendix G.5). A randomized check of Lemma 2 over 140 trials in five parameter sets satisfying Assumption 1, plus 30 targeted trials with cancelling pairs, found no incorrect share and no run above $k_D + f_D$ reads. These checks test the specification, not the implementation or the encryption of [30].

Baselines and coverage. Recovery with difference records needs Lk_D valid partials after L missed transitions, snapshots need k_D, and re-issuance needs an active committee (Table 5). On the implementation we did not measure possession-proof checks, localization, disputes, aborts, ledger crashes, wide-area networks, or repeated trials.

7 Conclusion

We have formalized where a failed refresh's published information must be charged, enforced this conservative rule with a ledger that is never rolled back, and justified it on the actual transcript. The committee is fixed, long-lived keys bound the offline workload, failure paths have been systematically validated on an executable reference model, and deploying a production BLS12-381 instantiation alongside an incentive analysis remains promising future work.

References

1. Abe, M., Fehr, S.: Adaptively secure Feldman VSS and applications to universally-composable threshold cryptography. In: Advances in Cryptology — CRYPTO 2004. LNCS, vol. 3152, pp. 317–334. Springer (2004)

2. Baron, J., Defrawy, K.E., Lampkins, J., Ostrovsky, R.: Communication-optimal proactive secret sharing for dynamic groups. In: Applied Cryptography and Network Security — ACNS 2015. LNCS, vol. 9092, pp. 23–41. Springer (2015)
3. Bellare, M., Boldyreva, A., Staddon, J.: Randomness re-use in multi-recipient encryption schemes. In: Public-Key Cryptography — PKC 2003. LNCS, vol. 2567, pp. 85–99. Springer (2003)
4. Bellare, M., Garay, J.A., Rabin, T.: Fast batch verification for modular exponentiation and digital signatures. In: Advances in Cryptology — EUROCRYPT 1998. LNCS, vol. 1403, pp. 236–250. Springer (1998)
5. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC). pp. 1–10. ACM (1988)
6. Benhamouda, F., Gentry, C., Gorbunov, S., Halevi, S., Krawczyk, H., Lin, C., Rabin, T., Reyzin, L.: Can a public blockchain keep a secret? In: Theory of Cryptography — TCC 2020. LNCS, vol. 12550, pp. 260–290. Springer (2020)
7. Blakley, G.R.: Safeguarding cryptographic keys. In: Proceedings of the 1979 AFIPS National Computer Conference. AFIPS Conference Proceedings, vol. 48, pp. 313–318. AFIPS Press (1979)
8. Boneh, D., Partap, A., Rotem, L.: Proactive refresh for accountable threshold signatures. In: Financial Cryptography and Data Security — FC 2024, Part II. LNCS, vol. 14745, pp. 140–159. Springer (2024)
9. Buterin, V., Feist, D., Loerakker, D., Kadianakis, G., Garnett, M., Taiwo, M., Dietrichs, A.: EIP-4844: Shard blob transactions. Ethereum Improvement Proposals (2022)
10. Cachin, C., Guerraoui, R., Rodrigues, L.: Introduction to Reliable and Secure Distributed Programming. Springer, 2 edn. (2011)
11. Cachin, C., Kursawe, K., Lysyanskaya, A., Strobl, R.: Asynchronous verifiable secret sharing and proactive cryptosystems. In: Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS). pp. 88–97. ACM (2002)
12. Cascudo, I., David, B.: Publicly verifiable secret sharing over class groups and applications to DKG and YOSO. In: Advances in Cryptology — EUROCRYPT 2024, Part V. LNCS, vol. 14655, pp. 216–248. Springer (2024)
13. Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Advances in Cryptology — CRYPTO 1992. LNCS, vol. 740, pp. 89–105. Springer (1993)
14. Chor, B., Goldwasser, S., Micali, S., Awerbuch, B.: Verifiable secret sharing and achieving simultaneity in the presence of faults. In: Proceedings of the 26th Annual Symposium on Foundations of Computer Science (FOCS). pp. 383–395. IEEE (1985)
15. Desmedt, Y., Jajodia, S.: Redistributing secret shares to new access structures and its applications. Tech. Rep. ISSE-TR-97-01, George Mason University (1997)
16. Ethereum Foundation: Ethereum proof-of-stake consensus specifications: Deneb networking interface. `https://github.com/ethereum/consensus-specs` (2024), parameter `MIN_EPOCHS_FOR_BLOB_SIDECARS_REQUESTS`
17. Ethereum Foundation Protocol Coordination Team: Fusaka mainnet announcement. Ethereum Foundation Blog (2025), `https://blog.ethereum.org/2025/11/06/fusaka-mainnet-announcement`
18. Faz-Hernández, A., Scott, S., Sullivan, N., Wahby, R.S., Wood, C.A.: Hashing to elliptic curves. RFC 9380, IRTF (2023)
19. Feist, D., D'Amato, F., Wang, H.W., et al.: EIP-7594: PeerDAS — peer data availability sampling. Ethereum Improvement Proposals (2024)

20. Feldman, P.: A practical scheme for non-interactive verifiable secret sharing. In: Proceedings of the 28th Annual Symposium on Foundations of Computer Science (FOCS). pp. 427–437. IEEE (1987)
21. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Advances in Cryptology — CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer (1987)
22. Francati, D., Venturi, D.: Evolving secret sharing revisited: Computational security and succinctness. Designs, Codes and Cryptography **93**, 3809–3861 (2025)
23. Fuchsbauer, G., Plouviez, A., Seurin, Y.: Blind Schnorr signatures and signed ElGamal encryption in the algebraic group model. In: Advances in Cryptology — EUROCRYPT 2020. LNCS, vol. 12106, pp. 63–95. Springer (2020)
24. Gennaro, R., Goldfeder, S.: Fast multiparty threshold ECDSA with fast trustless setup. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS). pp. 1179–1194. ACM (2018)
25. Gennaro, R., Jarecki, S., Krawczyk, H., Rabin, T.: Secure distributed key generation for discrete-log based cryptosystems. Journal of Cryptology **20**(1), 51–83 (2007)
26. Gennaro, R., Rabin, M.O., Rabin, T.: Simplified VSS and fast-track multiparty computations with applications to threshold cryptography. In: Proceedings of the 17th Annual ACM Symposium on Principles of Distributed Computing (PODC). pp. 101–111. ACM (1998)
27. Gentry, C., Halevi, S., Lyubashevsky, V.: Practical non-interactive publicly verifiable secret sharing with thousands of parties. In: Advances in Cryptology — EUROCRYPT 2022, Part I. pp. 458–487. LNCS, Springer (2022)
28. Goyal, V., Kothapalli, A., Masserova, E., Parno, B., Song, Y.: Storing and retrieving secrets on a blockchain. In: Public-Key Cryptography — PKC 2022. LNCS, vol. 13177, pp. 252–282. Springer (2022)
29. Groth, J.: Non-interactive distributed key generation and key resharing. Cryptology ePrint Archive, Paper 2021/339 (2021), `https://eprint.iacr.org/2021/339`
30. Groth, J., Shoup, V.: Design and analysis of a distributed ECDSA signing service. Cryptology ePrint Archive, Paper 2022/506 (2022), revised February 2023. `https://eprint.iacr.org/2022/506`
31. He, Y., Kodera, Y., Nogami, Y., Huda, S.: Role-based efficient proactive secret sharing with user revocation. Cryptography **9**(4), 80 (2025)
32. Herzberg, A., Jarecki, S., Krawczyk, H., Yung, M.: Proactive secret sharing or: How to cope with perpetual leakage. In: Advances in Cryptology — CRYPTO 1995. LNCS, vol. 963, pp. 339–352. Springer (1995)
33. Kalodner, H., Goldfeder, S., Chen, X., Weinberg, S.M., Felten, E.W.: Arbitrum: Scalable, private smart contracts. In: 27th USENIX Security Symposium. pp. 1353–1370. USENIX Association (2018)
34. Kate, A., Zaverucha, G.M., Goldberg, I.: Constant-size commitments to polynomials and their applications. In: Advances in Cryptology — ASIACRYPT 2010. LNCS, vol. 6477, pp. 177–194. Springer (2010)
35. Kivinen, T., Kojo, M.: More modular exponential (MODP) Diffie-Hellman groups for Internet key exchange (IKE). RFC 3526, IETF (2003)
36. Komargodski, I., Naor, M., Yogev, E.: How to share a secret, infinitely. In: Theory of Cryptography Conference (TCC) 2016-B. LNCS, vol. 9986, pp. 485–514. Springer (2016)
37. Kondi, Y., Magri, B., Orlandi, C., Shlomovits, O.: Refresh when you wake up: Proactive threshold wallets with offline devices. In: 2021 IEEE Symposium on Security and Privacy (SP). pp. 608–625. IEEE (2021)

38. Laing, T.M., Stinson, D.R.: A survey and refinement of repairable threshold schemes. Journal of Mathematical Cryptology **12**(1), 57–81 (2018)
39. Lambić, D.: Efficient non-interactive key refresh with multiple independent refreshers for threshold cryptography. Cryptology ePrint Archive, Paper 2026/839 (2026), https://eprint.iacr.org/2026/839
40. Mackey, M., et al.: EIP-7892: Blob parameter only hardforks. Ethereum Improvement Proposals (2025)
41. Maram, S.K.D., Zhang, F., Wang, L., Low, A., Zhang, Y., Juels, A., Song, D.: CHURP: Dynamic-committee proactive secret sharing. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS). pp. 2369–2386. ACM (2019)
42. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Advances in Cryptology — CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer (1991)
43. Pedersen, T.P.: A threshold cryptosystem without a trusted party. In: Advances in Cryptology — EUROCRYPT 1991. LNCS, vol. 547, pp. 522–526. Springer (1991)
44. Schnorr, C.P., Jakobsson, M.: Security of signed ElGamal encryption. In: Advances in Cryptology — ASIACRYPT 2000. LNCS, vol. 1976, pp. 73–89. Springer (2000)
45. Schoenmakers, B.: A simple publicly verifiable secret sharing scheme and its application to electronic voting. In: Advances in Cryptology — CRYPTO 1999. LNCS, vol. 1666, pp. 148–164. Springer (1999)
46. Schultz, D., Liskov, B., Liskov, M.: MPSS: Mobile proactive secret sharing. ACM Transactions on Information and System Security **13**(4), 34:1–34:32 (2010)
47. Shamir, A.: How to share a secret. Communications of the ACM **22**(11), 612–613 (1979)
48. Stadler, M.: Publicly verifiable secret sharing. In: Advances in Cryptology — EUROCRYPT 1996. LNCS, vol. 1070, pp. 190–199. Springer (1996)
49. Teutsch, J., Reitwießner, C.: A scalable verification solution for blockchains. arXiv:1908.04756 (2019)
50. Vassantlal, R., Alchieri, E., Ferreira, B., Bessani, A.: COBRA: Dynamic proactive secret sharing for confidential BFT services. In: 2022 IEEE Symposium on Security and Privacy (SP). pp. 1335–1353. IEEE (2022)
51. Vlasov, A., Olson, K., Stokes, A., Sanso, A.: EIP-2537: Precompile for BLS12-381 curve operations. Ethereum Improvement Proposals (2020)
52. Wahrstätter, T., Buterin, V., Dietrichs, A.: EIP-7623: Increase calldata cost. Ethereum Improvement Proposals (2024)
53. Wong, T.M., Wang, C., Wing, J.M.: Verifiable secret redistribution for archive systems. In: First International IEEE Security in Storage Workshop (SISW 2002). pp. 94–105. IEEE (2002)
54. Yurek, T., Xiang, Z., Xia, Y., Miller, A.: Long live the honey badger: Robust asynchronous DPSS and its applications. In: 32nd USENIX Security Symposium. pp. 5413–5430. USENIX Association (2023)
55. Zhou, L., Schneider, F.B., van Renesse, R.: APSS: Proactive secret sharing in asynchronous systems. ACM Transactions on Information and System Security **8**(3), 259–286 (2005)

A Notation and Formal Definitions

Table 3. Recurring notation.

Symbol	Meaning
$\lambda, \mathbb{G}, q, g, h$	security parameter; group, order, generators
e, s, r_0	epoch index; the secret; the masking constant preserved across epochs
$\mathcal{D}, D_j, n_D, k_D, f_D$	dealer committee, dealer, size, threshold, active-fault bound
$z_j, [x]$	dealer coordinate; degree-(k_D-1) sharing of x
$\mathcal{P}, \mathcal{P}_e, i, t_e$	participant set, epoch-e participants, identifier, epoch threshold
$P^{(e)}, R^{(e)}, \mathbf{C}^{(e)}$	epoch sharing and masking polynomials; commitment vector
A_σ, M_σ	bivariate dealer-side sharing and masking state of state σ
$C_{j,\ell}^{(e)}, \Delta_{j,\ell}, E_{j,i}$	per-dealer, delta, and evaluation commitments
$\mathsf{psh}_{j,i}^{(e)}, \mathsf{pupd}_{j,i}^{(e)}$	dealer j's enrollment partial and partial update token
$\chi_{j,i}, \pi^{\mathsf{epk}}$	common ciphertext component; proof of possession of the ephemeral key
α	$\log_g h$, known to no party and used only in the analysis
$\mathsf{aid}, \mathcal{F}_{\mathsf{Reg}}$	activation identifier; registry
$\mathcal{Q}_{\mathsf{gen}}, \mathcal{Q}_D^{(e)}$	qualified generation set; accepted per-dealer vectors
$Q_{\mathsf{gen}}, Q_{\mathsf{bc}}$	numbers of sharing and reliable-broadcast instances
$H_{\mathsf{batch}}, q_B, q_H$	batch-challenge oracle; number of batch tests; adversarial queries to H_{batch}
σ, a, ν	state (committed epoch or candidate); transition attempt; attempt nonce
$\mathcal{R}_e, \mathcal{O}_\sigma, \mathcal{O}_a$	transition recipients; participants moved onto σ, and onto the target of a
U_σ^{rel}	recipients of σ actually served by the public path
$\widehat{U}_\sigma$	release ledger of σ: union of $\mathcal{O}_a$ over the attempts that obtained a reservation
$\mathcal{D}_\sigma^{\mathsf{cor}}$	dealers corrupted while σ exists
$\mathcal{P}_\sigma^{\mathsf{cor}}$	participants whose corruption, whenever it occurs, exposes a σ-share, a private partial evaluation of σ at that participant, or staged material with σ as source or target
$\mathcal{E}_{P,\sigma}$	effective participant exposure of σ, see (1)
$C_{\mathsf{tr}}, Z_\sigma$	coordinates at which a subsharing message of a transition is exposed; $\mathcal{D}_\sigma^{\mathsf{cor}} \cup C_{\mathsf{tr}}^{\mathsf{in}}(\sigma)$
Π_Z	the polynomial of (12) that equals 1 at 0 and vanishes on Z
$\delta, \rho^{\mathsf{out}}, m$	corruption budget; outgoing reservation; publication margin $n_D - 2f_D - k_D$

A.1 Private Channels

Every private delivery, that is enrollment partials, direct tokens, recovery reissuance and the dealer-to-dealer channels of a generation subprotocol, runs over a forward-secret channel whose session state is erased on completion, and ϵ_{sync} is the advantage of distinguishing all of them from ideal confidential channels. Forward secrecy is a strict security requirement rather than an operational convenience: without it an archived ciphertext together with a later CorruptKey would recover past partials and defeat Theorem 2.

A.2 Formal Experiments

Definition 2 (Correctness and lifecycle integrity). *Assume that all parties follow the protocol and that the required messages are delivered.* Correctness *requires the following: combining the partials of participant i yields $(P^{(e)}(i), R^{(e)}(i))$; applying an update yields $(P^{(e+1)}(i), R^{(e+1)}(i))$;* Verify *accepts both against the respective public states;* $\mathsf{Reconstruct}$ *returns s from any t_e valid shares; and a participant retains no share of an earlier epoch once it has observed commit and completed the update. A participant in a pending enrollment holds dealer partials and no share at all, which is why $\mathcal{P}_\sigma^{\mathsf{cor}}$ charges partials rather than shares.* Lifecycle integrity *requires that* $\mathsf{BadIssue} \vee \mathsf{BadCons} \vee \mathsf{BadTrans}$ *have probability* $\mathsf{negl}(\lambda)$ *against f_D actively deviating dealers. The first two events are violations of authorized one-time issuance and of consistency among accepted shares and their reconstructions;* $\mathsf{BadTrans}$ *occurs exactly when two target states are simultaneously canonical, an aborted candidate is applied, one transition is applied twice, or a target state is installed without an accepted commitment record. A canonical state one epoch ahead of a participant that has not yet observed commit is* not *a violation; the discipline of Appendix B.2 leaves that participant holding its source share and staged token.*

Definition 3 (Proactive secrecy). *In $\mathsf{ExpPS}_{\Pi,\mathcal{A}}$ an admissible $\mathcal{A}$ chooses s_0, s_1 and a lifecycle schedule, the challenger draws b and runs Π on s_b for polynomially many finalized epochs, and $\mathcal{A}$ guesses b. The scheme provides proactive secrecy if $\mathsf{Adv}^{\mathsf{ps}}_{\Pi,\mathcal{A}} = |\Pr[\mathsf{ExpPS} = 1] - \frac{1}{2}|$ is negligible for every admissible $\mathcal{A}$.*

Definition 4 (Forward security: past-share unforgeability). *In $\mathsf{ExpFwd}_{\Pi,\mathcal{A}}$ the challenger runs Π against an admissible $\mathcal{A}$ through finalized epochs $0, \ldots, E$, then hands $\mathcal{A}$ the complete dealer state vector $\mathbf{st}_D^{(E)}$; $\mathcal{A}$ outputs $(\sigma^\star, i^\star, \mathsf{sh}^\star)$ and wins exactly when $\sigma^\star$ is a committed epoch earlier than E, $i^\star \in \mathcal{P}_{\sigma^\star} \setminus \mathcal{E}_{P,\sigma^\star}$, $|\mathcal{E}_{P,\sigma^\star}| \le t_{\sigma^\star} - 2$, and $\mathsf{sh}^\star$ is accepted by (5) for $\sigma^\star$. The scheme provides forward security if this probability is negligible.*

Definition 5 (Token privacy under a complaint oracle). *In $\mathsf{ExpTokPriv}_{\Pi,\mathcal{A}}$ the adversary controls at most f_D dealers with their signing keys and corrupts participants subject to Definition 1. It names an attempt (e, ν) and a public recipient $i^\star \in \mathcal{O}_a$ subject to neither* $\mathsf{CorruptParticipant}$ *nor* $\mathsf{CorruptKey}$ *at any point. For every honest dealer D_j let $m_j^{(0)} = \mathsf{pupd}_{j,i^\star}$ and $m_j^{(1)} = 0^{|m_j^{(0)}|}$; the challenger draws $b \in \{0,1\}$ and publishes, for every honest D_j, the record of* (10) *encrypting $m_j^{(b)}$ under $\mathsf{ad}_{j,i^\star}$, after which each dealer erases its ephemeral randomness, while corrupted dealers publish what $\mathcal{A}$ chooses. The challenger computes every subsequent action of $i^\star$, including recovery, acknowledgments and complaints, from $m_j^{(0)}$ regardless of b. The adversary may query $\mathsf{Dec}(i, \cdot)$ and $\mathsf{Cmp}(i, \cdot)$, the latter returning the verifiable decryption an honest recipient would publish, on every anchored record addressed to an honest participant $i \neq i^\star$ and on every anchored record addressed to $i^\star$ whose signer is a corrupted dealer, and on no challenge record. The advantage is $\mathsf{Adv}^{\mathsf{tok}}_{\Pi,\mathcal{A}} = |\Pr[\mathsf{ExpTokPriv}_{\Pi,\mathcal{A}} = 1] - \frac{1}{2}|$.*

The bound $t_{\sigma^\star} - 2$ rather than $t_{\sigma^\star} - 1$ in Definition 4 is necessary: the released state determines $s = P^{(\sigma^\star)}(0)$, so the secret acts as one further evaluation of the target polynomial, and an adversary that holds $t_{\sigma^\star} - 1$ exposed shares can interpolate the polynomial and forge the target share.

B Model and Release Accounting: Supplementary Proofs

B.1 Lemma 1: the Remaining Cases

Fix σ and let $\widehat{U}_\sigma^{\mathsf{fin}}$ be the final value of its ledger. A released attempt is in particular a reserved one, so $U_\sigma^{\mathsf{rel}} \subseteq \widehat{U}_\sigma^{\mathsf{fin}}$ and (1) gives $|\mathcal{E}_{P,\sigma}| \le \delta + |\mathcal{O}_\sigma \cup \widehat{U}_\sigma^{\mathsf{fin}}|$. If some attempt from σ passed step (iii), the last such attempt a extended the ledger to $\widehat{U}_\sigma \cup \mathcal{O}_a$ before any token existed and the ledger never shrinks, so that bound equals $\delta + |\mathcal{O}_\sigma \cup \widehat{U}_\sigma \cup \mathcal{O}_a| < t_\sigma$ by the source condition a passed. If none did, $\widehat{U}_\sigma^{\mathsf{fin}} = \varnothing$ and the target pre-check of the attempt that created σ, or the condition imposed at initialization, bounds $\delta + |\mathcal{O}_\sigma|$ below t_σ since $\rho^{\mathsf{out}} \ge 0$. Two cases remain.

A candidate abandoned at step (iii). If σ is the candidate of an attempt whose reservation failed, no token onto σ was ever computed, so $\mathcal{O}_\sigma = \varnothing$ and $U_\sigma^{\mathsf{rel}} = \varnothing$, whence $|\mathcal{E}_{P,\sigma}| \le |\mathcal{P}_\sigma^{\mathsf{cor}}| \le \delta < t_\sigma$ by the target pre-check of step (i), which that attempt passed before the candidate was generated. The strict sequential ordering of the five steps is specifically designed to address this case: if the candidate were generated before the target pre-check, a rejected attempt could still leave a state whose budget the lemma does not bound.

A crash between reservation and release. The reservation persists the extended ledger before returning, and no honest dealer computes or sends a token before the certificate exists, so a crash at any point in between leaves $\widehat{U}_\sigma$ already extended and U_σ^{rel} no larger than it would otherwise have been. Both bounds in the displayed chain of the main proof are therefore unaffected, and the same holds for a dealer that replays its durable log on recovery, since a replayed reservation is idempotent under the compare-and-set of Appendix B.4. Finally, the epoch-0 state has $\mathcal{O}_{\sigma_0} = \widehat{U}_{\sigma_0} = \varnothing$ and the corresponding condition is imposed at initialization. □

B.2 Material-Indexed Exposure Accounting

Two executions show why $\mathcal{P}_\sigma^{\mathsf{cor}}$ is defined by material rather than by time.

A staged token with a retained source share. Let $t = 2$ and $\delta = 1$, and write $P^{(0)}(x) = s + ax$ for the source epoch polynomial. The adversary corrupts participant 1 while the source state is current, obtaining $P^{(0)}(1)$, and then releases it. Participant 2 receives its update token over the direct path, stages it durably, acknowledges it, and crashes before applying it; the acknowledgment satisfies $q_2^{\mathsf{sync}} \ge k_D$ in (8), so the committee commits and erases the source dealer state. The adversary then corrupts participant 2, whose local state still contains $P^{(0)}(2)$, and computes $2P^{(0)}(1) - P^{(0)}(2) = 2(s + a) - (s + 2a) = s$. Counted

by time intervals, each state experienced at most one participant corruption and $|\mathcal{E}_{P,\sigma_0}| = 1 < 2 = t$, so Definition 1 would hold while s is recovered. Counted by material, the second corruption returns a source share, so participant 2 also belongs to $\mathcal{P}^{\mathsf{cor}}_{\sigma_0}$ and $|\mathcal{E}_{P,\sigma_0}| = 2 = t$, which Definition 1 excludes. The staged token alone is a difference and reveals nothing; the attack uses the pair of source share and staged token.

Single dealer partials. Let $k_D = t_\sigma = 2$ with dealer coordinates $z_1 = 1$ and $z_2 = 2$, and let four participants be corrupted during a pending enrollment, each holding one dealer partial: $A_\sigma(1,1)$, $A_\sigma(1,2)$, $A_\sigma(2,3)$ and $A_\sigma(2,4)$. None of them can form a share, since that needs $k_D = 2$ partials. But $\deg_x A_\sigma < 2$ gives $A_\sigma(1,0) = 2A_\sigma(1,1) - A_\sigma(1,2)$ and $A_\sigma(2,0) = 4A_\sigma(2,3) - 3A_\sigma(2,4)$, and $\deg_z A_\sigma(\cdot,0) < k_D$ then gives $s = A_\sigma(0,0) = 2A_\sigma(1,0) - A_\sigma(2,0)$. A definition that charges only participants holding complete shares would leave $\mathcal{E}_{P,\sigma}$ empty here; ours charges all four, giving $|\mathcal{E}_{P,\sigma}| = 4 \geq t_\sigma$. It is also the reading Lemma 3 needs, since J_σ vanishes on the whole column of every charged participant, so an exposed partial $A_\sigma(z_j, i)$ is preserved exactly when i is charged.

Participant-side discipline. The bound $|\mathcal{P}^{\mathsf{cor}}_\sigma| \leq \delta$ cannot be discharged by having a participant install the target share and erase the source before acknowledging: an attempt may abort after the acknowledgment, for instance when another recipient never answers, and the committee then keeps σ while that participant holds only σ', so a local atomic write does not give global commit atomicity. The protocol therefore separates preparation from application, and Theorem 1 is stated for this discipline.

Event	**Participant-side handling**
Prepare	retain the source share, stage the token durably, acknowledge
Commit observed	verify the target, apply once, erase source share and token
Abort observed	erase the token only, retain the source share

B.3 Sufficiency and Necessity of Cumulative Accounting

Proposition 1 (The bound is sufficient, not necessary). *In the variant of the protocol that drops the first condition of* (2) *and keeps only the per-attempt cap, there is an execution with $t_\sigma = 4$, one corrupted participant and three aborted attempts of target threshold 4 publishing to three distinct single recipients, in which $|\mathcal{E}_{P,\sigma}| = t_\sigma$, so Definition 1 is violated, and yet for every $\Delta \in \mathbb{Z}_q$ an assignment of the honest randomness leaves every commitment, every corrupted share and every published difference unchanged while replacing s by $s + \Delta$.*

Proof (of Proposition 1). Let $\sigma = \sigma_0$ be the initial state, so that no earlier state has to be shifted. Take n_D, k_D, f_D satisfying Assumption 1, no dealer corruption, $t_\sigma = 4$, and $\mathcal{P}^{\mathsf{cor}}_\sigma = \{4\}$, so $\delta = 1$. Let three attempts a_1, a_2, a_3 from σ have target threshold 4 and public-recipient sets $\mathcal{O}_{a_k} = \{k\}$, each releasing its tokens and then aborting. Each satisfies the per-attempt cap, $\delta + |\mathcal{O}_{a_k}| + \rho^{\mathsf{out}} = 1+1+0 < 4$, which is the only guard the variant of the statement retains. Under the full rule the third attempt would be refused, since its source condition reads

$\delta + |\mathcal{O}_\sigma \cup \widehat{U}_\sigma \cup \mathcal{O}_{a_3}| = 1 + |\{1,2,3\}| = 4 \not< 4$; this is why the proposition is stated for the variant, and it is also what Lemma 1 guarantees. In the variant, (1) gives $\mathcal{E}_{P,\sigma} = \{4\} \cup \{1,2,3\}$, so $|\mathcal{E}_{P,\sigma}| = 4 = t_\sigma$ and Definition 1 is violated.

The adversary's view consists of the commitment vector of σ and of each candidate, the share $(P^{(\sigma)}(4), R^{(\sigma)}(4))$ of the corrupted participant, and, for each k, the published difference $\big(\widehat{P}_{a_k}(k) - P^{(\sigma)}(k),\ \widehat{R}_{a_k}(k) - R^{(\sigma)}(k)\big)$ together with its evaluation commitment and ciphertext. Put $Q := \Pi_{\{4\}}$, of degree 1, and fix $\Delta \in \mathbb{Z}_q$. Apply to the source *and to every candidate* the same shift, replacing each sharing polynomial A by $A + \Delta Q$ and each masking polynomial M by $M - \alpha^{-1}\Delta Q$, and realize it on the dealer-shared state by the construction of Lemma 3 with the z-factor $\Pi_\varnothing \equiv 1$, which is admissible because no dealer is corrupted.

Degrees are respected, since $\deg Q = 1 < 4$ on both sides. Every commitment is unchanged by the identity in the proof of Lemma 3. The corrupted participant's share is unchanged because $Q(4) = 0$. Each published difference is unchanged because the *same* ΔQ was added to the source and to the candidate, so the two contributions cancel:

$$\big(\widehat{P}_{a_k} + \Delta Q\big)(k) - \big(P^{(\sigma)} + \Delta Q\big)(k) = \widehat{P}_{a_k}(k) - P^{(\sigma)}(k),$$

and likewise for the masking part, so the evaluation commitments and the ciphertexts, whose randomness is untouched, are unchanged as well. Yet $Q(0) = 1$, so the constant term of the source and of every candidate becomes $s + \Delta$, consistently. The view is therefore identical for every Δ. □

The witness is specific to public recipients that are not otherwise exposed: Q vanishes only on $\mathcal{P}_\sigma^{\mathsf{cor}}$, and an adversary that also held the source shares of $1, 2, 3$ would force a polynomial of degree less than 4 to vanish at four points while being nonzero at 0, which is impossible. The charge of (1) assumes that such an adversary may exist; it therefore over-approximates the knowledge of an adversary that never obtains the recipients' keys (Remark 1). The witness is also not available to Lemma 3, whose shift is chosen per state and must agree with the shifts of the neighboring states on the published differences, whereas the witness uses the fact that all three attempts aborted from one source.

Necessity of cumulative accounting. The two executions of Section 1 correspond to the two guards of (2). The first, a transition to a lower threshold whose tokens two corrupted participants combine before the attempt aborts, is a genuine reconstruction of s; such an execution exists whenever the candidate threshold is at most the number of participants that hold both a source share and a usable candidate token. The target pre-check of step (i) rejects it, because $\delta + |\mathcal{O}_a| + \rho_a^{\mathsf{out}} = 2 + 0 + 0 \not< 2 = t_a$. The second, three aborted attempts that publish to three distinct single recipients, is not a reconstruction: Proposition 1 gives a witness under which the adversarial view is consistent with every secret. It shows that a per-attempt cap does not enforce the accumulated bound of Definition 1, since each attempt satisfies $\delta + |\mathcal{O}_a| + \rho_a^{\mathsf{out}} = 1 + 1 + 0 < 4$

while $|\mathcal{E}_{P,\sigma}|$ reaches 4; the source condition of step (iii), evaluated against the accumulated ledger, refuses the third attempt. Neither execution shows that (1) is the tightest accounting: a policy that tracks the linear dependencies among published differences could admit more executions, and Proposition 3 indicates what such an analysis would have to handle.

B.4 The Release Ledger

Assumption 2 (Authentication layer). *Dealers hold two independent signature key pairs, one for records, enrollment attestations and commitment vectors, and one for reservation certificates. At every point of the execution the reservation key of a dealer that is not currently corrupted cannot be used by the adversary.*

Two instantiations satisfy Assumption 2. Either the reservation key resides in a trust domain that CorruptDealer does not reveal, or the authentication layer is itself subject to mobile compromise, in which case a recovering dealer registers fresh keys through a procedure anchored outside the committee and every signed object carries the authentication epoch of its signer. In the second case, admission of a new object accepts only the key epoch currently registered for the signer, so a retired key cannot create a new reservation, record or attestation, whereas verification of an already finalized object uses the registry snapshot fixed at its finalization, so a participant returning within the retention window still verifies a record whose signer has since rotated its key. Non-frameability (Theorem 5) additionally requires that the signer's record key was not exposed while it was current. A signature authenticates origin, not the honesty of its author or the correctness of its plaintext; the dispute path of Section 5 attributes the latter.

Table 4. Authentication dependencies. "Fresh" is a condition at the moment the object was created and admitted, not when it is later read.

Object	Key	Condition	Used by
Reservation certificate	reservation	fresh, quorum $n_D - f_D$	Lemma 1
Per-dealer commitment vector	record	fresh	Thm. 4, Thm. 7
Issuance attestation	record	fresh	Thm. 4, Lem. 6
Update record	record	fresh	Thm. 3, Thm. 5, Thm. 6
Epoch update transaction	chain	anchor binding	Thm. 6, Thm. 7

$\mathsf{ReserveRelease}(\sigma, \mathsf{root}_{\mathsf{old}}, \nu, \mathcal{O}_a)$ is a compare-and-set: it succeeds only if the ledger of σ still has root $\mathsf{root}_{\mathsf{old}}$, in which case it appends $\mathcal{O}_a$, persists the result and returns $\mathsf{root}_{\mathsf{new}}$. Reliable broadcast alone does not implement this. Reliable broadcast guarantees that honest parties deliver the same messages, not that they deliver different messages in the same order, so two dealers may deliver two proposals in opposite orders and each let a different reservation succeed. We therefore require the dealers to run a totally ordered log, which any Byzantine

state-machine replication protocol provides [10], and treat reservations as commands of that log; a deployment may instead sequence reservations by blockchain finality, at the cost of one finalized transaction per attempt.

A reservation is released only against a certificate of $n_D - f_D$ dealer signatures on $(\sigma, \mathsf{version}, \mathsf{root}_{\mathsf{old}}, \nu, \mathcal{O}_a, \mathsf{targetDigest}, \mathsf{root}_{\mathsf{new}})$, and an honest dealer signs at most one tuple per $(\sigma, \mathsf{version})$. Two certificates for the same version share at least $n_D - 2f_D \geq k_D > f_D$ signers by Assumption 1, hence at least one dealer that is honest and whose signing key the adversary cannot use, so under Assumption 2 they cannot carry different $\mathcal{O}_a$: safety of the compare-and-set follows from the quorum intersection and the locking rule, while liveness, view change, and recovery follow standard state-machine replication guarantees.

No honest dealer computes or sends a public token before the certificate exists, so a crash between the certificate and the release leaves the ledger already extended and the released set smaller, which is the conservative direction; a dealer recovering from a crash replays the durable log before acting. Reclassifying a non-acknowledging participant to the public path within an attempt is itself a ReserveRelease whose added set is that participant, so a reclassification the rule rejects leaves no trace.

B.5 Finalization and Progress

The commit condition is (8), evaluated only after the reservation of step (iii) has succeeded. A participant classified as online that does not acknowledge within the staging window is reclassified as a public recipient of the same attempt: the committee re-evaluates (2) with $\mathcal{O}_a$ extended by that participant and either extends the ledger and publishes a record or abandons the attempt. Without this rule, a single corrupted participant that answers the liveness check and then withholds its acknowledgment would stall every attempt, and prolonged use of one state weakens proactive security. Condition (2) is a sufficient policy rather than a characterization: committing also requires the generation subprotocol to terminate, the qualified set to be agreed, pending activations to resolve, and the epoch update to reach finality. When the source condition fails, the failure is permanent for that state, since t_σ is fixed and past publications cannot be withdrawn; only the target threshold and the reservations of future states remain free.

Each state passes through $\mathsf{generated} \to \mathsf{staged} \to \mathsf{committed}$ or $\mathsf{aborted} \to \mathsf{erased}$. Commit anchors the record, makes the target state current, applies each durably staged direct token exactly once, and erases the source coefficient shares, the attempt's temporary sharing state and every plaintext partial. Abort erases the candidate coefficient shares, the staged tokens and the attempt's ephemeral randomness, but not the release ledger, and a retry uses a fresh nonce, so no candidate polynomial is reused. Erasure is a core protocol requirement rather than an implementation detail: Lemma 3 uses it to identify the coordinates that ever observe a transition message, and Theorem 2 to bound what a released dealer state contains. For the minimal committees $n_D = 3f_D + 1$, $k_D = f_D + 1$ of Section 6, the margin of (9) is $m = 0$.

C Security Analysis: Full Proofs

C.1 Mixed Record Formats

Corollary 1 (Which state a public record is charged to). *Let each attempt publish either difference records or records that carry the target partial, and replace* $\mathcal{E}_{P,\sigma}$ *in* (1) *by* $\mathcal{E}^{\mathsf{mix}}_{P,\sigma} := \mathcal{P}^{\mathsf{cor}}_{\sigma} \cup \mathcal{O}_{\sigma} \cup U^{\mathsf{rel,diff}}_{\sigma}$*, where the last union runs over the released attempts from* σ *that used difference records, with the source condition of* (2) *and the reservation ledger restricted to those attempts. Then Lemma 3 and Theorem 1 hold verbatim with* $\mathcal{E}^{\mathsf{mix}}$ *in place of* $\mathcal{E}$*. We claim this for these two results only; the dispute and evaluation results are stated for the difference format.*

Proof (of Corollary 1). Inspect the proof of Lemma 3 step by step. The exposure sets, the constant coefficient, the resharing corrections (15) and (16), the higher coefficients, the commitment identity and the bijection argument never mention the format of a public record; the only step that does is the one about public update tokens. There, a difference record published for i at the transition from σ to σ' has plaintext $A_{\sigma'}(z_j, i) - A_{\sigma}(z_j, i)$ and masking counterpart, which the shift leaves unchanged exactly when $J_{\sigma'}(z, i) - J_{\sigma}(z, i) = 0$ for every z. Our shifts are chosen state-locally and independently, so making both vanish is a simple sufficient condition covering every admissible execution, and $i \in U^{\mathsf{rel,diff}}_{\sigma} \cap \mathcal{O}_{\sigma'}$ gives it by (17); it is not necessary, since Proposition 2 links the two shifts and preserves a difference with neither vanishing. A record carrying $A_{\sigma'}(z_j, i)$ instead has that value as its plaintext, which is unchanged as soon as $J_{\sigma'}(z, i) = 0$; the target charge $i \in \mathcal{O}_{\sigma'}$ already gives this, and the source shift J_{σ} is free to be nonzero at $x = i$. Hence charging the source only for attempts that published differences suffices, and $\mathcal{E}^{\mathsf{mix}}_{P,\sigma}$ is the set on which J_{σ} must vanish. The degree bound needed in (17) becomes $\deg_x J_{\sigma} = |\mathcal{E}^{\mathsf{mix}}_{P,\sigma}| < t_{\sigma}$, which is the modified admissibility condition, and Lemma 1 carries over verbatim once $\widehat{U}_{\sigma}$ accumulates only the difference attempts, since its proof uses nothing about $\mathcal{O}_a$ beyond set inclusion. Theorem 1 then applies unchanged. □

The evaluation commitments and ciphertexts follow their plaintexts in both cases, since the encryption randomness is untouched. Note that the corollary does not make the two formats interchangeable at a fixed budget: it establishes which state each format obliges the accounting to charge, and Table 5 summarizes the resulting trade-offs and operational overheads incurred by each format.

C.2 Shift Invariance: the Omitted Verifications

Rationale for the common exposure set. Under the synchronous round model $C_d = C_{\mathsf{tr}}$ for every $d \notin B$, where $C_d \subseteq C_{\mathsf{tr}}$ denotes the coordinates at which D_d's subsharing is exposed, because a dealer corrupted during the transition receives a subshare from every contributor and $R_d \subseteq \mathcal{D}^{\mathsf{cor}}_{\sigma} \cap \mathcal{D}^{\mathsf{cor}}_{\sigma'}$; so correcting each contributor on its own C_d would coincide with (15). The distinction matters

as soon as one asks whether (13) alone suffices, that is, whether the argument survives a relaxation of synchrony in which $C_d \subsetneq C_{\mathsf{tr}}$ is possible. The argument holds under the common correction of (15), but fails under a per-contributor correction: if $w \in R_h \setminus C_j$ for some $j \neq h$, the aggregate term of (16) evaluates to $-\widetilde{j}_{\sigma'}(z_w)/\lambda_h^S(0) \neq 0$, so the value $V_h(z_w)$ that the complaint answer already published would change, and since h is selected by a public rule an adversary can direct a complaint at exactly that contributor. The common product in (15) removes the dependence on C_d, so (13) is the only hypothesis Lemma 3 uses. The union in (13) ranges over every honest contributor, selected or not, and over both subprotocols of the transition: every executed honest contribution is corrected, so an opening of an unselected contributor at a coordinate outside C_{tr} would otherwise change a published answer.

The higher coefficients. With J_σ as in (17) we have $\deg_z J_\sigma < k_D$, $\deg_x J_\sigma = |\mathcal{E}_{P,\sigma}| < t_\sigma$, $J_\sigma(z,0) = j_\sigma(z)$, $J_\sigma(z_u, x) \equiv 0$ for $u \in Z_\sigma$ and $J_\sigma(z,i) \equiv 0$ for $i \in \mathcal{E}_{P,\sigma}$. Hence $J_\sigma^+ := J_\sigma - j_\sigma$ also vanishes on Z_σ, so replacing the contribution of h' leaves every subshare a corrupted dealer received from h' unchanged, and $h' \notin B$ because $B \subseteq \mathcal{D}_\sigma^{\mathsf{cor}} \subseteq Z_\sigma$. A coordinate opened by a complaint during a joint sharing is therefore accounted for exactly as one opened during a resharing, since $C_{\mathsf{tr}}^{\mathsf{in}}(\sigma) \subseteq Z_\sigma$ by definition.

Complete inductive argument. The corrections are computed from sets known only once the execution has ended, so the map is defined on the honest randomness r, namely r_0 and the free coefficients of every honest sharing, contribution and masking polynomial, *through* the execution that r produces. Fix the adversary's random tape, and write $\mathrm{Sets}(r;s)$ for the tuple $(\mathcal{D}_\sigma^{\mathsf{cor}}, \mathcal{P}_\sigma^{\mathsf{cor}}, \mathcal{O}_\sigma, \widehat{U}_\sigma, C_{\mathsf{tr}}, B, S, h, h')$ over all states of the execution on secret s, $\mathrm{View}(r;s)$ for the adversary's view, and $\Phi_\Delta^s(r)$ for the randomness obtained by applying the corrections of Lemma 3, computed from $\mathrm{Sets}(r;s)$; the encryption and channel randomness is untouched.

Claim. $\mathrm{View}(\Phi_\Delta^s(r); s+\Delta) = \mathrm{View}(r;s)$, $\mathrm{Sets}(\Phi_\Delta^s(r); s+\Delta) = \mathrm{Sets}(r;s)$, and $\Phi_{-\Delta}^{s+\Delta} \circ \Phi_\Delta^s = \mathrm{id}$.

Proof. Induct on the sequence of adversary-observable events: a broadcast or published object, a message delivered to a corrupted party, a corruption response, and a complaint answer. Suppose the first k events coincide in the two executions. The adversary's next action is a deterministic function of its tape and of those events, so it coincides; in particular it corrupts the same party, raises the same complaint, and withholds the same acknowledgment. The next honest event is of one of three kinds. A commitment or broadcast object is unchanged by the identity in the proof of Lemma 3. A private value at a coordinate the adversary observes lies, by the induction hypothesis, in one of the corrupted or opened sets of $\mathrm{Sets}(r;s)$, on which the corresponding correction vanishes by construction, so the delivered value is unchanged. An honest decision, whether to complain, whether to accept a sharing, whether a batch test accepts (which under the

hypothesis of Lemma 3 depends only on residuals fixed by the corrupted dealers' messages and public commitments), the choice of S, the qualified set of a joint sharing, and the evaluation of (2), depends only on verification outcomes and public data, which are unchanged; in particular an honest recipient's shifted pair still opens the unchanged commitment, so it raises no new complaint, and the deterministic rules that select S, h and h' return the same choices. Hence event $k+1$ coincides, which proves the first two statements, the second because the sets are functions of the adversary's actions and the honest decisions. For the third, every correction is linear in Δ for fixed sets by (14), (15), (16) and (17), and the sets coincide, so applying the construction with $-\Delta$ returns r. □

Consequently Φ^s_Δ is a bijection of the finite set of honest randomness vectors. Each coordinate it replaces is a free uniform coordinate, namely a non-constant coefficient of an honest sharing, a constant term contributed to a joint sharing, or r_0, while a coordinate determined by the program, such as the constant term $V_d(0) = x_d$ of a resharing, is moved exactly to the value the shifted program prescribes. Hence Φ^s_Δ preserves the distribution of the honest randomness, and $\mathrm{View}(\cdot; s)$ and $\mathrm{View}(\cdot; s+\Delta)$ are identically distributed.

C.3 Forward Security: the Incident Objects

We verify that the shift (18) of Theorem 2 leaves every object of the execution unchanged. The objects and the reason each is preserved are as follows; σ^- denotes the state from which $\sigma^\star$ was created.

Object	**Reason**
disclosed final dealer state, s, r_0	$J(z,0) \equiv 0$ and no other state is shifted
corrupted rows of $\sigma^\star$ and opened coordinates	$\Pi_{Z_{\sigma^\star}}$ vanishes there
material returned at an exposed column	$\Pi_{\mathcal{E}_{P,\sigma^\star}}$ vanishes there
public tokens of the incoming transition and of every outgoing attempt *that published*	each recipient lies in $\mathcal{O}_{\sigma^\star} \cup U^{\mathsf{rel}}_{\sigma^\star}$ by (1)
commitments, $\Delta_{j,\ell}$, $E_{j,i}$	the Pedersen cancellation identity

The winning event. Let $(\sigma^\star, i^\star, \mathsf{sh}^\star)$ be the output and $\widehat{\mathsf{sh}}$ the share the protocol generated. When the experiment returns 1 both open $\mathbf{C}^{(\sigma^\star)}(i^\star)$. On the event that they differ, the adversary exhibits two openings of one Pedersen commitment, so that event has probability at most ϵ_{bind}; the remainder of the argument conditions on their being equal.

Degrees. $\deg_z J = |Z_{\sigma^\star}| \le k_D - 1$ by (13), and $\deg_x J \le p+1 \le t_{\sigma^\star} - 1$, where $p = |\mathcal{E}_{P,\sigma^\star}|$. The extra unit of x-degree is the price of the factor x that makes $J(z,0) \equiv 0$, and it is why the experiment budgets $t_{\sigma^\star} - 2$.

The released state. Since $J(z,0) \equiv 0$, the constant coefficients of $A_{\sigma^\star}$ and $M_{\sigma^\star}$ are untouched, so s and r_0 are preserved and so is every later state, including the disclosed $\mathbf{st}^{(E)}_D$. In particular the resharing that produced $\sigma^\star$ from σ^- and the resharing that produces σ^+ from $\sigma^\star$ are unaffected, because both act on

the constant coefficient only; the shift touches only the coefficients that JRSS generated inside $\sigma^\star$, through the contributor h' chosen as in Lemma 3.

Rows, columns and opened coordinates. $\Pi_{Z_{\sigma^\star}}$ vanishes on $\mathcal{D}^{\mathsf{cor}}_{\sigma^\star}$ and on $C^{\mathsf{in}}_{\mathsf{tr}}(\sigma^\star)$, so every corrupted row of $\sigma^\star$ and every subsharing value opened during the transition from σ^- is preserved; $\Pi_{\mathcal{E}_{P,\sigma^\star}}$ vanishes on $\mathcal{E}_{P,\sigma^\star}$, so every exposed column is preserved.

Commitments and records. Every commitment is preserved by the identity in the proof of Lemma 3. Hence so is every difference $\Delta_{j,\ell}$ incident to $\sigma^\star$, being a quotient of two of them, and so is every evaluation commitment $E_{j,i}$. The public update tokens of the incoming transition and of *every* outgoing attempt that published are addressed to recipients in $\mathcal{O}_{\sigma^\star} \cup U^{\mathsf{rel}}_{\sigma^\star} \subseteq \mathcal{E}_{P,\sigma^\star}$, on which J vanishes, so their plaintexts and therefore their ciphertexts are unchanged. An attempt that reserved without publishing leaves no public token to preserve, its recipients lying in $\widehat{U}_{\sigma^\star}$ but not necessarily in $U^{\mathsf{rel}}_{\sigma^\star}$, and its private material falls under the corrupted-row and exposed-column cases above; the direct tokens are hidden by the ideal channels of the first game transition.

Conclusion. $J(0, i^\star) = \gamma$ by (12) and the choice of the normalizing denominator, so the map shifts $P^{(\sigma^\star)}(i^\star)$ by γ and $R^{(\sigma^\star)}(i^\star)$ by $-\alpha^{-1}\gamma$. Conditioned on the whole view, $\widehat{\mathsf{sh}}$ is uniform over q equally likely values, so the conditional success probability is $1/q$. Adding the binding term gives the bound of Theorem 2.

C.4 Token Privacy: the Reduction

Proof (of Theorem 3). $\mathcal{B}$ plays the associated-data-only chosen-ciphertext game against a challenge recipient $i^\star$. If $i^\star$ is fixed in advance, $\mathcal{B}$ registers it through the challenger's interface for registering an honest user and publishes the returned key as its enrollment key; otherwise it guesses $i^\star$ among the N participants and aborts on a wrong guess, incurring a security degradation factor of N. Every other participant key pair is generated by $\mathcal{B}$, so an adaptive corruption of any participant other than $i^\star$ is answered with the corresponding secret key, and the game forbids corrupting $i^\star$. $\mathcal{B}$ also generates all dealer signing keys and hands those of the at most f_D corrupted dealers to $\mathcal{A}$.

$\mathcal{B}$ runs the protocol as specified and encrypts every record itself using the public keys, with one exception: for every honest dealer D_j it submits $(m_j^{(0)}, m_j^{(1)})$ under $\mathsf{ad}_{j,i^\star}$ to the challenger's encryption oracle, which the game admits any polynomial number of times, and publishes the returned ciphertexts. The set of associated data ever submitted to that oracle is therefore exactly $\{\mathsf{ad}_{j,i^\star} : D_j \text{ honest}\}$. Routing the remaining records through the encryption oracle as well would make their associated data inadmissible for later decryption-proof queries; encrypting them locally circumvents this restriction without additional overhead, as encryption requires only public keys. Because $\mathcal{B}$ generates the execution, it knows the real tokens $m_j^{(0)}$ and computes the subsequent actions of $i^\star$ from them, as Definition 5 prescribes; when $i^\star$ processes or disputes a record signed by a corrupted dealer, $\mathcal{B}$ obtains the plaintext or the decryption proof from its own oracles, which admit such records by the argument below.

A query on a record addressed to a participant $i \neq i^\star$ is answered locally, because $\mathcal{B}$ holds that participant's secret key. A query on a record addressed to $i^\star$ and signed by a corrupted dealer D_c carries $\mathsf{ad}_{c,i^\star}$, which differs from every submitted associated datum, since c ranges over corrupted indices only; by Lemma 4 no other record carries that datum, so the queries are admissible and $\mathcal{B}$ forwards them. A record addressed to $i^\star$ that carries an honest dealer's index but differs from the record that dealer published fails admission unless a signature is forged, an event of probability at most ϵ_{sig}. Hence $\mathcal{B}$ never queries a challenge ciphertext, and it outputs the bit $\mathcal{A}$ outputs. Conditioned on a correct guess and on no forgery the simulation is perfect, so the stated bound follows. The reduction never needs the ephemeral exponent of a challenge record, which a reduction to plain chosen-ciphertext security of a generic scheme would require in order to produce that record's proof of possession, and it performs no extraction from adversarial proofs. □

Remark 2 (Why the proof of possession is required). The simpler instance of the scheme in [30], which omits π^{epk}, is analyzed relative to a sensibility predicate and admits a decryption-proof query only after a non-rejecting decryption query with the same associated data and common component. In our application the predicate is exactly "the plaintext opens $E_{j,i}$", and a complaint is a decryption proof on a record whose plaintext *fails* it. The instance carrying the proof of possession is therefore a requirement of the complaint interface and not a convenience.

C.5 Irrevocability of Complaint Disclosures

The dealer layer is synchronous, so an honest contributor answers only complaints raised by corrupted recipients and C_{tr} stays inside the corrupted set. The following execution shows why that must be supplied by the model in advance rather than checked afterwards. Let $k_D = 2$ and $f_D = 1$, let the source sharing be $X(z) = s + bz$, and let the adversary corrupt D_1, so that it holds $x_1 = X(1)$. Let the accepted resharers be $S = \{D_1, D_2\}$, with the honest D_2 resharing $x_2 = X(2)$ under $V_2(z) = x_2 + az$. The corrupted D_1 receives $V_2(1)$. If a complaint resolution additionally opens V_2 at the coordinate of an honest D_3, the adversary obtains

$$x_2 = \tfrac{1}{2}\big(3V_2(1) - V_2(3)\big), \qquad \text{hence} \qquad s = 2x_1 - x_2.$$

One direct corruption and one public answer at an honest coordinate therefore determine the secret, and $|C_{\mathsf{tr}} \cup \mathcal{D}^{\mathsf{cor}}_{\sigma'}| = |\{1,3\}| = 2 > k_D - 1$, so (13) rejects the configuration. A published answer cannot be revoked: abandoning the attempt leaves s reconstructible, exactly as an aborted transition leaves its released update tokens in the adversary's possession. With $k_D = f_D + 1$, as in every committee of Section 6, (13) leaves no room for any answer opened at an honest coordinate while f_D dealers are corrupted; this example is the case $f_D = 1$. A deployment that cannot guarantee synchrony must therefore keep (13)

true by its own rule: before broadcasting an answer, a contributor checks that the union of coordinates opened in the transition, including the new one, has at most $k_D - 1 - f_D$ elements, which bounds $|C_{\mathsf{tr}} \cup \mathcal{D}^{\mathsf{cor}}_{\sigma'}|$ by $k_D - 1$ whatever the corrupted set is, and otherwise stops without answering, which affects only liveness. Raising k_D alone does not suffice: with $k_D = 3$ and $f_D = 1$, a corrupted recipient's coordinate together with two answers at honest coordinates determines a degree-2 subsharing, and the same for several honest resharers determines s. The alternative is a resolution procedure that never opens an honest contributor's subshare in the clear, for example publicly verifiable dealing [29,27].

C.6 Key-Only Compromise and the Record Format

Because the dealers hold the target coefficients, a record could equally carry the target partial $(A_{\sigma'}(z_j, i), M_{\sigma'}(z_j, i))$, which lets a returning participant reconstruct its new share directly and recover from any number of missed epochs with one snapshot. Table 5 compares this format with difference records and with re-issuance. A target-partial record reveals the current share to any party that obtains sk_i, including an adversary that archives the log and obtains the key much later, whereas a difference record reveals only one linear relation between the source and target shares. The following two propositions make this comparison precise.

Table 5. Recovery designs for a participant that missed L transitions, at equal (n_D, k_D, f_D, t) and equal record encryption; the rows are consequences of the constructions, not measurements.

	Difference	**Snapshot**	**Re-issuance**
Dealer contact on return	none	none	required
Valid partials needed	Lk_D	k_D	k_D responses
Source share needed	yes	no	no
Source-side accumulation	yes, by (1)	no	no public record
A later key leak yields	one relation per transition	the target share	channel-dependent

Proposition 2 (Key-only compromise, isolated transition). *Let i be a public recipient of an attempt from σ to σ' whose long-term key the adversary holds through* CorruptKey *but whose share it holds in neither state, and suppose $|\mathcal{D}^{\mathsf{cor}}_{\sigma} \cup \mathcal{D}^{\mathsf{cor}}_{\sigma'}| < k_D$ for that transition. Consider the two states in isolation, that is, with no other public record addressed to i. Then the shift of Lemma 3 can be chosen so that the published difference for i is unchanged without requiring J_σ or $J_{\sigma'}$ to vanish at $x = i$; that is, i need not be counted in either exposure set for the argument to go through, at the cost of one unit of degree in the target threshold. If the record instead carried the target partial, no such choice exists: the plaintext is the target share, so the shift must vanish at i in σ' and i must be counted there.*

Proof. By hypothesis a single j of degree less than k_D vanishes on $\mathcal{D}^{\mathsf{cor}}_{\sigma} \cup \mathcal{D}^{\mathsf{cor}}_{\sigma'}$ with $j(0) = \Delta$, so both states may use the same dealer-side factor. Write the shifts as $J_\sigma(z, x) = j(z)R(x)$ and $J_{\sigma'}(z, x) = j(z)R'(x)$. Choose R of degree at most $|\mathcal{E}_{P,\sigma}|$ with $R(0) = 1$ vanishing on $\mathcal{E}_{P,\sigma}$, and then R' of degree at most $|\mathcal{E}_{P,\sigma'}| + 1$ with $R'(0) = 1$, vanishing on $\mathcal{E}_{P,\sigma'}$, and satisfying $R'(i) = R(i)$; the latter is $|\mathcal{E}_{P,\sigma'}| + 2$ interpolation conditions at distinct points and is solvable whenever $t_{\sigma'} > |\mathcal{E}_{P,\sigma'}| + 1$. The published plaintext for i is $A_{\sigma'}(z_j, i) - A_\sigma(z_j, i)$, whose shift is $j(z_j)(R'(i) - R(i)) = 0$, and the masking difference is shifted by $-\alpha^{-1}$ times the same quantity, so the record is unchanged. Every other requirement of Lemma 3 is met because R and R' vanish on the respective exposure sets and j vanishes on both corrupted dealer sets. For the target-partial variant the plaintext is $A_{\sigma'}(z_j, i)$ itself, whose shift is $j(z_j)R'(i)$; since j is nonzero at some honest coordinate, fixing the plaintext forces $R'(i) = 0$, which is the statement that i is counted in $\mathcal{E}_{P,\sigma'}$. □

The hypothesis on the union of adjacent corrupted dealer sets is stronger than anything Lemma 3 needs, and (1) does not rely on Proposition 2: it charges every public recipient. The proposition identifies the condition under which difference records are preferable to target-partial records at a single transition.

Proposition 3 (The isolated-transition argument does not compose). *Proposition 2 does not extend to a chain of transitions, even when its hypothesis holds at every adjacent pair. Concretely, let $k_D = 3$, let the states be $\sigma_0 \to \sigma_1 \to \sigma_2$ with $\mathcal{D}^{\mathsf{cor}}_{\sigma_0} = \{1\}$, $\mathcal{D}^{\mathsf{cor}}_{\sigma_1} = \{2\}$ and $\mathcal{D}^{\mathsf{cor}}_{\sigma_2} = \{3\}$, so that every adjacent union has size $2 < k_D$, and let i be served by the public path at both transitions. An adversary that archives the records and later calls $\mathsf{CorruptKey}(i)$ recovers $P^{(\sigma_2)}(i)$ and $R^{(\sigma_2)}(i)$.*

Proof. Write $x_{e,j} = A_{\sigma_e}(z_j, i)$ and let $d_{e,j} = x_{e+1,j} - x_{e,j}$ be the plaintext of the record from D_j to i at transition e, which the adversary obtains from the archived ciphertexts once it holds sk_i. The dealer corruptions give $x_{0,1}$, $x_{1,2}$ and $x_{2,3}$. Then $x_{2,1} = x_{0,1} + d_{0,1} + d_{1,1}$ and $x_{2,2} = x_{1,2} + d_{1,2}$, so all three dealer coordinates of the column of i in σ_2 are known and $P^{(\sigma_2)}(i) = \sum_j \lambda_j(0) x_{2,j}$ follows by interpolation; the masking column is recovered identically. In general, fix (j, i) and form the graph whose vertices are the states and whose edges are the transitions for which a record from D_j to i was published and decrypted. Knowing one absolute partial propagates over the whole connected component, and the share is recovered at any state at which k_D coordinates become known. □

This execution is not an attack on Theorem 1, because (1) charges i to both endpoints of every transition at which it is a public recipient. It shows instead that the conservative charge cannot be dropped by iterating Proposition 2, which compares two record formats at one transition and is not a guarantee over a lifecycle. A lifecycle statement under key-only compromise would have to track the propagation graph above; we leave its formalization open.

D Protocol Details, Correctness, and Recovery

D.1 The Generation Subprotocols

Rather than introducing a novel verifiable secret sharing scheme, both subprotocols rely on standard Pedersen verifiable sharing with complaint handling over authenticated point-to-point channels and reliable broadcast, in the sense of [42,25], utilizing the following modular interface. A dealer D_d invokes $\mathsf{Share}(m, r, \tau)$, which broadcasts coefficient commitments $C_0, \ldots, C_{\tau-1}$ and delivers a pair to each dealer; a recipient that finds $g^v h^\rho \neq \prod_\ell C_\ell^{z^\ell}$ broadcasts a complaint, and D_d answers by broadcasting the disputed pair. The properties we invoke are: *agreement*, that all honest dealers output the same qualified set; *consistency*, that a qualified sharing is a consistent degree-$(\tau-1)$ sharing except with probability ϵ_{VSS}; and *shift compatibility*, that for every Δ and every polynomial W of degree less than τ with $W(0) = \Delta$ that vanishes on the coordinates listed in (13), replacing the sharing polynomial by $A+W$ and the masking polynomial by $M - \alpha^{-1}W$ leaves every published commitment and every message an exposed party sees unchanged.

The published $C_0 = g^m h^r$ determines (m, r) up to the line $\{(m + \Delta,\, r - \alpha^{-1}\Delta)\}$, so we do not assume that the view of at most $\tau - 1$ recipients is independent of (m, r); the analysis of Section 4 uses the line, which also covers $\mathsf{ReShare}$, where the mask is inherited from the source sharing rather than drawn afresh. We assume $f_D < n_D/3$ and reliable broadcast with failure probability ϵ_{bc}; adaptive security of such sharings is studied in [1].

$\mathsf{ReShare}([x])$. Every D_j invokes $\mathsf{Share}(x_j, \rho_j, k_D)$ on its own share. A sharing is accepted when it is qualified and its constant-term commitment equals the publicly computable commitment to x_j; those two checks are broadcast-decided, so all honest dealers accept the same set $\mathcal{Q}$. If $|\mathcal{Q}| < k_D$ the attempt aborts, which (P2) makes impossible when no honest dealer crashes, since $n_D - f_D \geq k_D$. Honest dealers take $S \subseteq \mathcal{Q}$ to be the k_D smallest indices, a deterministic rule, and each D_u sets its new share to $\sum_{j \in S} \lambda_j^S(0) x_{j,u}$; the new commitments follow by the same combination in the exponent. Public messages are the commitments, the complaints and the answers; the individual pairs are otherwise private. A complaint answer publishes one pair, so the set of coordinates at which a contributor's subsharing is exposed is not in general the set of corrupted dealers: an honest contributor answering a spurious complaint opens its subshare at the complainant's coordinate. Under synchronous delivery within the round an honest contributor's message reaches every honest recipient, so only a corrupted recipient can complain against it and the two sets coincide; without that assumption the opened coordinates must be counted, which (13) does.

$\mathsf{JRSS}(\nu, \ell)$. Every D_d samples (u_d, w_d) and invokes $\mathsf{Share}(u_d, w_d, k_D)$ with the domain separator (ν, ℓ). The qualified set $\mathcal{Q}_{\mathsf{gen}}$ is again broadcast-decided, the attempt aborts if $|\mathcal{Q}_{\mathsf{gen}}| < n_D - f_D$, and the outputs are the componentwise sums over $\mathcal{Q}_{\mathsf{gen}}$. We make no distributional assumptions regarding the sum: Lemma 3

leaves adversarial contributions untouched and requires only that at least one contributor in $\mathcal{Q}_{\mathsf{gen}}$ was never corrupted while the state exists, which (P2) guarantees. Specifically, the phrase "fresh sharing" denotes a sharing produced by a distinct subprotocol invocation with an updated domain separator, rather than implying that its coefficients are uniformly distributed.

Completion and erasure. A subprotocol completes when its qualified set is decided and its outputs are computed. At that point every dealer erases its contribution and subsharing polynomials, the pairs it received, and the randomness of the invocation, retaining only its own share of the output. Lemma 3 uses exactly this boundary: a dealer corrupted after completion yields only a row of the state, whereas one corrupted during the subprotocol lies in C_{tr}.

D.2 The Encryption Instance

Encryption under associated data ad samples κ, sets $v = g^{\kappa}$, derives $g' = H_{\mathbb{G}}(\mathsf{epk\text{-}pop}, \mathsf{ad}, v)$, sets $v' = (g')^{\kappa}$, and attaches $\pi^{\mathsf{epk}} = (v', \mathsf{PoK}\{\kappa : v = g^{\kappa} \wedge v' = (g')^{\kappa}\})$; the message key is $H_{\mathsf{M}}(\mathsf{derive\text{-}key}, \mathsf{ad}, i, \mathsf{pk}_i, v, \mathsf{pk}_i^{\kappa})$ and the individual component is $c = m + H_{\mathsf{M}}(\cdot)$ over $\mathbb{Z}_q^2$. The common component is $\chi = (v, \pi^{\mathsf{epk}})$, of size $32 + 32 + 64 = 128$ bytes at $\lambda = 128$, and the individual component is 64 bytes; the proof of possession is an equality of discrete logarithms over a hash-derived base and occupies 96 bytes rather than the 64 of a plain proof of knowledge. The instance is associated-data-only chosen-ciphertext secure under an interactive Diffie–Hellman assumption in the random-oracle model [30].

D.3 Anchoring and Record Authentication

The anchored log of an attempt is encoded as a sequence of 31-byte field elements filling one or more blobs; the epoch update transaction stores the versioned hashes of those blobs. A record is admitted on chain by supplying its byte string together with an opening that binds it to a stored versioned hash. Two encodings are compatible with the point-evaluation precompile of [9], which verifies one evaluation of the blob polynomial. In the first, the claimant supplies the field elements of the record and one precompile invocation per element, which authenticates the record directly at a cost linear in its length. In the second, which the modeled figures of Section 6 use, the blob's first field element carries the root of a Merkle tree over record digests; a single precompile invocation authenticates the root, and an inclusion proof of depth $\lceil \log_2 N \rceil$ for N records in the blob authenticates the digest, after which the record bytes are checked against it. In either case, the cryptographic binding from the on-chain versioned hash to the full record tuple $(e, e{+}1, \nu, j, i, \mathsf{ct}, E_{j,i}, \pi^{\mathsf{epk}}, \sigma)$ is strictly preserved.

The consistency dispute of Appendix E.4 needs a second, finer interface, since it authenticates a single coefficient commitment $C_{j,\ell}^{(\sigma)}$ rather than a whole record. A per-dealer commitment vector is therefore committed as a Merkle tree over the coefficient-indexed leaves $(\ell, C_{j,\ell})_{\ell<t}$, whose root is the leaf placed in the

record tree, so one coefficient opening costs $\lceil \log_2 t \rceil + \lceil \log_2 N \rceil$ hashes above the single precompile invocation, ten plus the record depth at $t = 1024$; a flat record digest would instead force the whole vector to be supplied. At most two such openings, one per state, enter the final step, and the qualification is necessary when the threshold falls: with $t_\sigma = 3$ and $t_{\sigma'} = 2$ the disputed index $\ell^\star = 2$ has a leaf in the source tree and none in the target tree, even though the padded value there is $1_{\mathbb{G}}$. The verifier therefore evaluates

$$\mathsf{GetCommitment}(\sigma, j, \ell) = \begin{cases} \text{the value of a verified opening}, & \ell < t_\sigma, \\ 1_{\mathbb{G}}, & t_\sigma \le \ell < T, \end{cases}$$

with t_σ read from the authenticated state header, and demands an opening only in the first branch; a deployment may instead commit both vectors padded to T, at the cost of $T - t_\sigma$ extra leaves. Without this branch a dispute would be unresolvable exactly when the threshold decreases, a case the protocol otherwise supports.

The source vectors must therefore remain retrievable throughout the entire dispute lifecycle of any transition originating from that state, which is stronger than the single-window condition of Section 3: writing $T_{\mathsf{disp}} = T_{\mathsf{challenge}} + 2\lceil \log_2 T \rceil T_{\mathsf{round}} + T_{\mathsf{response}} + T_{\mathsf{finality}}$, a deployment must satisfy $T_{\mathsf{disp}} < T_{\mathsf{retention}}$ and re-anchor, or archive, the source per-dealer commitment vectors at each transition. The aggregate $\mathbf{C}^{(e)}$ discussed as long-lived verification state is not the same object as the per-dealer $C_{j,\ell}^{(e)}$ this dispute consumes. Returning to the record path, the modeled 93,347 gas covers one precompile invocation and the root check only; the inclusion proof and the signature check are the separately reported calldata and signature costs.

D.4 The Registry State Machine

Each identifier and public key follows $\mathsf{unused} \to \mathsf{pending} \to \mathsf{issued}$, with two further transitions: a pending activation not completed before a deadline moves to $\mathsf{expired}$ and no longer blocks a transition, and an issued registration may move to $\mathsf{recover\text{-}pending}$ and back when a returning participant holds fewer than k_D valid update records, in which case the dealers evaluate the current epoch polynomial exactly as in an enrollment. Recovery therefore needs no retained previous-epoch state, and the delivered value equals the one the update path would have produced. Conflicting reuse of an identifier or key is rejected, and a transition finalizes only when no source-epoch activation is pending. Table 6 gives the algorithm signatures.

D.5 Batch Verification and Its Soundness

The participant checks the signatures and, after decrypting every partial, derives in canonical dealer order the challenges

$$\eta_j = H_{\mathsf{batch}}(\mathsf{BATCH} \,\|\, \mathsf{aid} \,\|\, e \,\|\, i \,\|\, \mathsf{tr} \,\|\, j),$$
$$\mathsf{tr} = \big(\mathsf{Pub}^{(e)}, \mathsf{aid}, i, (j, \mathsf{att}_{j,i}^{(e)}, v_{j,i}^{(e)}, \rho_{j,i}^{(e)})_{j \in S}\big),$$

Table 6. Protocol syntax. aid is an activation identifier and π_i^{reg} a registration authorization issued by the enrollment policy. $\mathsf{ShareInit}$ samples r_0 internally, uniformly and independently of s.

Algorithm	Signature
Setup	$(1^\lambda, t_0, k_D, n_D) \to \mathsf{pp}$
ShareInit	$(s, t_0) \to (\mathsf{Pub}^{(0)}, \mathbf{st}_D^{(0)})$
Enroll	$(\mathsf{aid}, e, i, \mathsf{pk}_i, \pi_i^{\mathsf{reg}}) \to (\mathsf{reg}_i, \{\mathsf{psh}_{j,i}^{(e)}\}_{j \in S_D})$ or $\perp$
PartialJoin	$(\mathbf{st}_{D_j}^{(e)}, \mathsf{reg}_i) \to \mathsf{psh}_{j,i}^{(e)}$
CombineShare	$(\mathsf{Pub}^{(e)}, \mathsf{reg}_i, \{\mathsf{psh}_{j,i}^{(e)}\}_{j \in S_D}) \to \mathsf{sh}_i^{(e)}$ or $\perp$
Verify	$(\mathsf{Pub}^{(e)}, i, \mathsf{sh}_i^{(e)}) \to \{\mathsf{acc}, \mathsf{rej}\}$
Transition	$(e, t_{e+1}, \mathcal{R}_e, \nu, \mathbf{st}_D^{(e)}) \to (\widehat{\mathsf{Pub}}^{(e+1)}, \widehat{\mathbf{st}}_D^{(e+1)}, \{\mathsf{pupd}_{j,i}^{(e)}\})$
Finalize	$b \in \{\mathsf{commit}, \mathsf{abort}\}$
ApplyUpdate	$(e, i, \mathsf{sh}_i^{(e)}, \{\mathsf{pupd}_{j,i}^{(e)}\}_{j \in S_D}) \to \mathsf{sh}_i^{(e+1)}$ or $\perp$
Reconstruct	$(e, \{\mathsf{sh}_i^{(e)}\}_{i \in \mathcal{T}}) \to s$ or $\perp$

and accepts the batch when

$$g^{\sum_{j \in S} \eta_j v_{j,i}^{(e)}} h^{\sum_{j \in S} \eta_j \rho_{j,i}^{(e)}} \stackrel{?}{=} \prod_{j \in S} \prod_{\ell < t_e} \left(C_{j,\ell}^{(e)}\right)^{\eta_j i^\ell}, \tag{19}$$

which batches the per-dealer relation $g^{v_{j,i}^{(e)}} h^{\rho_{j,i}^{(e)}} = C_j^{(e)}(i)$. A failed batch is split recursively until the invalid responses are isolated, and the same technique verifies m shares in one multi-exponentiation rather than m.

where $H_{\mathsf{batch}} : \{0,1\}^* \to \mathbb{Z}_q$ is modeled as a random oracle. An implementation instantiates it with the hash-to-field method of [18] at security level λ, whose output is within statistical distance $2^{-\lambda}$ of uniform in $\mathbb{Z}_q$; this adds at most $k_D 2^{-\lambda}$ per batch to the bound below.

Lemma 5 (Batch soundness). *Let the challenges be derived as above, so that the canonical transcript* tr *fixes every value the residuals depend on before any challenge is drawn, and let* q_H *be the number of the adversary's queries to* H_{batch}*. A batch that contains at least one partial with a nontrivial residual is accepted with probability at most* $(q_H + 1)/q$*.*

Proof. For partial j define the residual $\Gamma_j := g^{v_{j,i}} h^{\rho_{j,i}} \left(\prod_\ell (C_{j,\ell})^{i^\ell}\right)^{-1}$, so (19) holds if and only if $\prod_{j \in S} \Gamma_j^{\eta_j} = 1_{\mathbb{G}}$. Because tr contains every $(v_{j,i}, \rho_{j,i})$ and every per-dealer commitment digest, the vector $(\Gamma_j)_{j \in S}$ is a function of the transcript. Write $\Gamma_j = g^{d_j}$; the batch accepts exactly when $\sum_j \eta_j d_j \equiv 0 \pmod q$. Fix a transcript with $d_{j^\star} \neq 0$. Its challenges are outputs of H_{batch} on distinct inputs, hence independent and uniform in $\mathbb{Z}_q$; conditioned on all η_j with $j \neq j^\star$, the equation holds for exactly one value of $\eta_{j^\star}$, so it holds with probability $1/q$. The transcript on which the batch is finally evaluated is either one on which the adversary queried H_{batch}, of which there are at most q_H, or an unqueried one; a union bound gives $(q_H + 1)/q$. □

The hypothesis on tr is structurally necessary rather than purely conservative. If the challenges were derived from the epoch digest and the scalar-free issuance

attestations alone, two corrupted dealers could read η_a, η_b first and submit $v'_a = v_a + d$ and $v'_b = v_b - \eta_a \eta_b^{-1} d$ for any $d \neq 0$: both residuals are non-trivial, yet $\Gamma_a^{\eta_a} \Gamma_b^{\eta_b} = g^{\eta_a d} g^{-\eta_a d} = 1_{\mathbb{G}}$ and the batch is accepted with certainty. Such a batch still yields a combined share that (5) rejects, so this is a failure of the batch test and of the recursive isolation built on it rather than of share consistency; binding the partials into tr mitigates this vulnerability without additional communication overhead, since the recipient already holds them and nothing about them is published.

D.6 Correctness, Lifecycle Integrity, and Robust Recovery

Theorem 4 (Correctness and lifecycle integrity). *The scheme satisfies Definition 2, the integrity part with advantage at most* $\epsilon_{\mathsf{reg}} + \epsilon_{\mathsf{fin}} + \epsilon_{\mathsf{sig}} + \epsilon_{\mathsf{bind}} + Q_{\mathsf{gen}}\epsilon_{\mathsf{VSS}} + Q_{\mathsf{bc}}\epsilon_{\mathsf{bc}} + q_B(q_H+1)/q$ *for* q_B *batch tests and* q_H *adversarial queries to* H_{batch}.

Here ϵ_{reg} and ϵ_{fin} bound failures of the registry state machine and of on-chain finalization, respectively.

Proof (of the correctness part). For coefficient index ℓ let $A_{e,\ell}$ and $M_{e,\ell}$ be the degree-(k_D-1) dealer polynomials sharing $a_\ell^{(e)}$ and $r_\ell^{(e)}$, so that $A_{e,\ell}(z_j) = a_{j,\ell}^{(e)}$ and $M_{e,\ell}(z_j) = r_{j,\ell}^{(e)}$. For any qualified S_D with $|S_D| \geq k_D$,

$$\sum_{j \in S_D} \lambda_j^{S_D}(0) v_{j,i}^{(e)} = \sum_{\ell=0}^{t_e-1} \Big(\sum_{j \in S_D} \lambda_j^{S_D}(0) A_{e,\ell}(z_j) \Big) i^\ell = \sum_{\ell=0}^{t_e-1} A_{e,\ell}(0) i^\ell = P^{(e)}(i),$$

because Lagrange interpolation of a polynomial of degree less than k_D at k_D points is exact. Replacing $A_{e,\ell}$ by $M_{e,\ell}$ gives $R^{(e)}(i)$, so $\mathsf{CombineShare}$ returns $(P^{(e)}(i), R^{(e)}(i))$. By (4), $\prod_\ell (C_\ell^{(e)})^{i^\ell} = g^{P^{(e)}(i)} h^{R^{(e)}(i)}$, so (5) accepts.

For the update, let $T = \max(t_e, t_{e+1})$ and let $\bar{a}_\ell^{(e)}$ be the source coefficients zero-padded to length T; padding does not change the polynomial. Interpolating k_D update-token partials gives $\delta v_i = \sum_{\ell<T} (\bar{a}_\ell^{(e+1)} - \bar{a}_\ell^{(e)}) i^\ell = P^{(e+1)}(i) - P^{(e)}(i)$, and likewise for the masking part, so $\mathsf{ApplyUpdate}$ returns $(P^{(e+1)}(i), R^{(e+1)}(i))$, and the same commitment computation for epoch $e+1$ shows that Verify accepts. The relation $\prod_{\ell<T} \Delta_\ell^{i^\ell} = g^{\delta v_i} h^{\delta \rho_i}$ used in (7) follows from $\Delta_\ell = C_\ell^{(e+1)} (C_\ell^{(e)})^{-1}$ with the same padding convention, and holds whether t_{e+1} exceeds, equals, or is smaller than t_e. Finally, for any $T^\star$ of size t_e consisting of identifiers whose shares pass (5), $\sum_{i \in T^\star} \lambda_i^{T^\star}(0) P^{(e)}(i) = P^{(e)}(0) = s$. The erasure clause holds by the rule of Section 2. □

Proof (of the integrity part). Let Fail be the union of a registry failure, a finalization failure, a signature forgery, a violation of Pedersen binding, a false batch acceptance, and a consistency failure of a Pedersen verifiable sharing or of reliable broadcast; a union bound gives the stated advantage. Condition on $\neg\mathsf{Fail}$.

Issuance. The registry state machine of Appendix D.4 admits only the transitions listed there, each requiring a valid authorization, and an issued registration can be re-entered only through recover-pending, which delivers an evaluation of the current epoch polynomial at the same identifier and therefore the same value the update path would have produced. A completion certificate consists of k_D unforgeable attestations bound to $(\mathsf{aid}, e, i, \mathsf{pk}_i)$. Hence BadIssue cannot occur.

Consistency. For an accepted share (v_i, ρ_i), (5) gives $g^{v_i} h^{\rho_i} = \prod_\ell (C_\ell^{(e)})^{i^\ell} =: C^{(e)}(i)$. Two distinct accepted shares for the same (e, i) would be two openings of $C^{(e)}(i)$, contradicting $\neg$Fail. For an accepted set $\mathcal{T}$ of size t_e put $s_\mathcal{T} = \sum_{i \in \mathcal{T}} \lambda_i^\mathcal{T}(0) v_i$ and $r_\mathcal{T} = \sum_{i \in \mathcal{T}} \lambda_i^\mathcal{T}(0) \rho_i$. Raising each verification equation to $\lambda_i^\mathcal{T}(0)$ and multiplying gives $g^{s_\mathcal{T}} h^{r_\mathcal{T}} = C_0^{(e)}$, because $\sum_{i \in \mathcal{T}} \lambda_i^\mathcal{T}(0) i^\ell$ equals 1 for $\ell = 0$ and 0 for $1 \le \ell < t_e$. Two accepted sets reconstructing distinct secrets would give two openings of $C_0^{(e)}$. Hence BadCons cannot occur.

Transitions. Under $\neg$Fail every accepted sharing is a consistent degree-$(k_D - 1)$ sharing and all honest dealers agree on the same qualified set, so ReShare outputs a sharing of the same constant term: the subsharing of D_j is accepted only if its constant-term commitment equals the publicly derived commitment to D_j's source share, and a dealer passing that check while resharing another value would open one Pedersen commitment twice. Lagrange combination over an accepted set of size k_D then preserves the constant term exactly, and JRSS contributes fresh sharings of the remaining coefficients, whose values the statement does not constrain. The target commitment record is accepted only under (3), which by Assumption 1 fixes the commitment polynomial of every coefficient index, so no unauthenticated target becomes canonical and no two are canonical at once; the canonical epoch record is installed on commit and not on abort, so an aborted candidate is never applied, and the epoch index with the attempt nonce makes a second application a duplicate. BadTrans therefore cannot occur. This concerns the committee's canonical record: a participant that has not observed commit still holds its source share and staged token, which Appendix B.2 permits and the erasure clause, conditioned on completing ApplyUpdate, does not contradict. □

Lemma 6 (Registry view independence). *The contents of $\mathcal{F}_{\mathsf{Reg}}$, including every completion certificate, are computable from the public state and the dealers' signing keys alone. The shift of Lemma 3 preserves both inputs, so it preserves the registry view; we do not claim that the registry is independent of the shares, only that it adds nothing to what the public state already fixes.*

Proof. The registry stores, per activation, the tuple $(\mathsf{aid}, e, i, \mathsf{pk}_i)$, its state, and, once issued, a set of at least k_D issuance attestations. An attestation is a signature on $(\mathsf{aid}, e, i, \mathsf{pk}_i, j, C_j^{(e)}(i))$, and $C_j^{(e)}(i) = \prod_{\ell < t_e} (C_{j,\ell}^{(e)})^{i^\ell}$ is a product of the anchored public commitments with public exponents i^ℓ. Every field is therefore a function of the epoch public state, the anchored commitment record, the participant identifier and public key, and the signing keys, all of which Lemma 3 leaves unchanged; the registry view is consequently the same in the shifted execution. □

Proof (of Lemma 2). Let $\mathcal{R}_i$ be the dealer-indexed record set that finalization fixed for i, so $|\mathcal{R}_i| \geq n_D - f_D$ by (8); let $\mathcal{B}_i \subseteq \mathcal{R}_i$ be the records that are not valid for i, so $|\mathcal{B}_i| \leq f_D$ because the record of an honest dealer passes all three tests; and let $\mathcal{M}_i$ be the valid records that i cannot retrieve, with $|\mathcal{M}_i| \leq m$ by hypothesis. Counting gives $|\mathcal{R}_i \setminus (\mathcal{B}_i \cup \mathcal{M}_i)| \geq (n_D - f_D) - f_D - m = k_D$, so enough valid records exist, and the authenticated states and per-dealer vectors that the hypothesis provides are what localization needs to evaluate (11). Every record the algorithm reads is either valid or in $\mathcal{B}_i$. *Reads.* If the algorithm terminates in the aggregate phase, it has kept exactly k_D records and discarded, before combining, only records that failed admission or opening, all of which lie in $\mathcal{B}_i$; it thus reads at most $k_D + |\mathcal{B}_i|$ records, although some kept records may also lie in $\mathcal{B}_i$. If it enters localization, every record it discards fails one of the three tests and lies in $\mathcal{B}_i$, and it stops once k_D valid records are kept, so again it reads at most $k_D + |\mathcal{B}_i|$. In both cases the number of reads is at most $k_D + f_D$; the minimum k_D counts valid partials, not fetches.

Termination in the aggregate phase. Suppose (7) accepts a combined pair $(\widetilde{\delta v}_i, \widetilde{\delta \rho}_i)$. Then $g^{\widetilde{\delta v}_i} h^{\widetilde{\delta \rho}_i} = \boldsymbol{\Delta}(i) = g^{\delta v_i} h^{\delta \rho_i}$, so a pair that differs from $(\delta v_i, \delta \rho_i)$ is a second opening of $\boldsymbol{\Delta}(i)$, which occurs with probability at most ϵ_{bind}; adding the source share to the correct pair gives the epoch-$(e{+}1)$ share, which opens $\mathbf{C}^{(e)}(i)\boldsymbol{\Delta}(i) = \mathbf{C}^{(e+1)}(i)$ as (5) requires. Acceptance does not certify the individual records: with $k_D = 3$, $f_D \geq 2$ and dealer coordinates $1, 2, 3$, the Lagrange coefficients at 0 are $(3, -3, 1)$, so two corrupted dealers that add the same $u \neq 0$ to their plaintexts and sign evaluation commitments that the modified plaintexts open both violate (11), while the combined token is unchanged.

Termination after localization. If (7) fails, localization discards every candidate that fails (11) and replaces it with a further record. The records it finally combines are valid, their Lagrange combination is $(\delta v_i, \delta \rho_i)$, and (7) accepts; the discarded records identify their dealers as faulty. □

E Dispute Resolution

A complaint that a record is malformed must publish a verifiable decryption, a stronger interface than a decryption oracle: the recipient reveals $K_{j,i} = \mathsf{epk}_{j,i}^{\mathsf{sk}_i}$ with an equality proof [13]. That interface is unsafe when ephemeral keys are unauthenticated, since a corrupted dealer that republishes an honest record's ephemeral key raised to a known exponent recovers, from the resulting complaint, the shared value of the honest record. The proof of possession carried by the scheme of Section 3 rules this out and belongs to that scheme rather than being an addition of ours [30]; what the lifecycle must supply is its query restriction, which Lemma 4 establishes, and Remark 2 explains why the variant without the proof of possession is not an option here.

We write D for decryption, DP and DV for generating and verifying a decryption proof, and DfP for recovering the plaintext from a verified decryption proof. For a component X, namely the equality proof DLEQ or the KZG point-evaluation opening [34,9], $\epsilon_X^{\mathsf{comp}}$, $\epsilon_X^{\mathsf{snd}}$ and ϵ_X^{zk} denote its completeness, soundness

and zero-knowledge errors; $\epsilon_{\mathsf{anchor}}$ bounds a break of anchor binding, and $\epsilon_{\mathsf{contract}}$ an incorrect contract execution.

E.1 Complaint Guarantees

Theorem 5 (Complaint guarantees). *Under the admission and plaintext-validity tests of Section 5, a valid complaint about an admitted malformed record is accepted except with probability $\epsilon^{\mathsf{comp}}_{\mathsf{DLEQ}}$; a complaint against a well-formed record signed by an honest dealer with an unexposed signing key is accepted with probability at most $\epsilon_{\mathsf{sig}} + \epsilon_{\mathsf{anchor}} + \epsilon^{\mathsf{snd}}_{\mathsf{KZG}} + \epsilon^{\mathsf{snd}}_{\mathsf{DLEQ}} + \epsilon_{\mathsf{bind}}$; and an object failing admission is rejected without any dealer being charged.*

Proof. Completeness. Admission is a deterministic re-execution of the anchor opening and the signature check, so an object that fails it is rejected with probability 1 and no dealer is charged; the subsequent analysis therefore restricts attention to admitted objects. For an admitted record whose proof of possession fails, attribution is again a deterministic re-execution and succeeds with probability 1. Otherwise the honest recipient runs DP, whose output DV accepts by completeness of the encryption scheme, except with probability $\epsilon^{\mathsf{comp}}_{\mathsf{DLEQ}}$ for the equality proof it contains, and DfP then returns exactly the plaintext D returns, again by completeness. By the definition of a malformed record that pair does not open $E_{j,i}$, so the complaint is accepted.

Non-frameability. Consider a well-formed record signed by an honest dealer whose signing key is unexposed, and define the games F_0, the frame experiment; F_1, aborting on a signature forgery; F_2, aborting on an anchor-binding or record-membership failure; F_3, aborting on a failure of soundness of the decryption proof, that is on a proof accepted by DV for which DfP does not return D's output; and F_4, aborting on a Pedersen binding failure. The successive differences are bounded by ϵ_{sig}, $\epsilon_{\mathsf{anchor}}+\epsilon^{\mathsf{snd}}_{\mathsf{KZG}}$, $\epsilon^{\mathsf{snd}}_{\mathsf{DLEQ}}$ and ϵ_{bind}. In F_4 the complaint is bound to the dealer-signed anchored record, the accepted decryption proof yields through DfP exactly the plaintext the dealer encrypted, and that plaintext opens $E_{j,i}$, which by binding is the only pair it opens. The verifier therefore rejects, so $\Pr[F_4 = 1] = 0$ and the stated bound follows. No step uses ciphertext integrity: a claimant that mauls a ciphertext produces an object that fails admission, since the dealer signature covers the record.

Secret-key hiding. Replace the equality proof by its zero-knowledge simulator. The remainder of the transcript, including the deliberately disclosed $K_{j,i}$, is unchanged, and the real and simulated transcripts differ by at most $\epsilon^{\mathsf{zk}}_{\mathsf{DLEQ}}$. Note that $K_{j,i}$ is disclosed only for a record whose proof of possession verifies, whose creator consequently already knows it. □

E.2 Availability-Challenge Attribution

Theorem 6 (Availability-challenge attribution). *Under anchor binding, opening soundness, and correct contract execution, a challenge resolves only with a record bound to the anchor fixed at finalization, and no dealer supplying a valid required opening before the deadline is penalized.*

Proof. Finalization fixes the obligation tuple $(e, e{+}1, \nu, j, i, \mathsf{anchor}, T_{\mathsf{response}})$. A response resolves the challenge only if it arrives before the deadline and the submitted record opens against anchor. Accepting data not bound to anchor would therefore require an anchor-binding violation or an opening-soundness violation, which bounds the first probability by $\epsilon_{\mathsf{anchor}} + \epsilon^{\mathsf{snd}}_{\mathsf{KZG}}$.

The same binding excludes the strategy of omitting a record, waiting for a challenge, and then revealing it against a retroactively chosen commitment. If the disputed record belongs to the finalized object, a valid opening exists and shows only that the record was published, not that it was retrievable when the claim was filed; such a challenge is resolved by the response and the challenger is reimbursed rather than treated as having accused falsely. If the dealer omitted it, then answering requires either an opening against a different commitment, which anchor binding rejects, or a false opening against the finalized commitment, which violates opening soundness. The omitted-record case therefore reduces to withholding.

If the dealer submits the required record with a valid opening before the deadline, completeness of the opening scheme makes verification accept and the contract enters the resolved state; penalizing the dealer would require a contract-state failure or a completeness failure, which bounds the second probability by $\epsilon_{\mathsf{contract}} + \epsilon^{\mathsf{comp}}_{\mathsf{KZG}}$ and gives zero for a correct deterministic contract and a complete opening scheme. Otherwise the deterministic deadline transition records that D_j failed to satisfy its publication obligation. This establishes public attribution; it does not establish that the data was retrievable at every point of the retention window. □

E.3 Availability Challenges and the Payout Rule

A recipient missing a record proves $i \in \mathcal{O}_{e+1}$ and deposits a bond $\mathcal{S}_C$. Dealer D_j must answer within T_{response} with an opening against the anchor fixed at finalization [9], or prove that no record was required; failure forfeits its stake. Because a response is accepted only against that anchor, a dealer cannot omit a record, await a challenge, and then reveal it against a commitment of its own choosing. Publication in the past and retrievability at the time of a challenge are distinct, so a valid opening does not show that the request was unfounded. Let $\mathcal{S}_D = \mathcal{W} + \mathcal{B}$ be the dealer stake, split into a finder reward and a burned part, and let the service fee be $F = F_{\mathsf{serv}} + F_{\mathsf{ext}}$, where F_{serv} is paid to the answering dealer and F_{ext} is burned. The contract distinguishes four outcomes and moves funds as follows.

Two features of this table matter. An honest requester and one that files a valid request solely to impose cost submit the same object and receive the same response, so no on-chain evidence distinguishes them and the bond cannot be forfeited in one case and returned in the other; charging F on every valid request prices the dealer's work without asserting an intent the contract cannot observe. And the default branch charges no fee, because no service was rendered.

Table 7. Settlement rules of the availability challenge.

Outcome	Settlement
Admission fails: invalid anchor proof or signature, or the claimant is not a recipient of the attempt	bond $\mathcal{S}_C$ forfeited to the dealer; a verdict about the claimant's evidence
Valid request, dealer answers in time	bond returned; requester charged F, of which F_{serv} goes to the dealer and F_{ext} is burned
Valid obligation, dealer does not answer	dealer default: $\mathcal{S}_D$ forfeited, $\mathcal{W}$ paid to the challenger, $\mathcal{B}$ burned, bond returned, no service fee charged
Repeat request for a record already served	rejected as a duplicate; no funds move

Scope of the settlement rule. The settlement rule fixes who bears which cost on which on-chain evidence, and Theorem 6 states the attribution it yields. It does not follow that withholding is unprofitable: whether attribution deters depends on an external model of gains and costs. One deviation illustrates the difficulty. A withholding dealer that observes an incoming challenge can file its own challenge from an identity it controls, so that the finder reward returns to it and only the burned part is lost. Consequently, our formal guarantees establish deterministic attribution and attributed non-response, while a comprehensive incentive model addressing withholding remains an open direction for future work. A forfeited stake must be replenished before the dealer serves again, and a monitor without the recipient's decryption key can detect the absence of a record but cannot judge whether a record is malformed, so the two monitoring roles are provisioned separately.

E.4 Adjudicating an Inconsistent Evaluation Commitment

The plaintext test of Section 5 decides whether the pair a recipient recovers opens $E_{j,i}$. It does not decide whether $E_{j,i}$ is the evaluation of the dealer's own delta vector: a corrupted dealer may select arbitrary $(\tilde{v}, \tilde{\rho})$, publish $\widetilde{E}_{j,i} = g^{\tilde{v}} h^{\tilde{\rho}}$, encrypt that pair honestly and sign the record, so that the plaintext test passes while (11) fails. Any party detects such an i off chain with the batched check

$$\prod_{i \in \mathcal{O}_{e+1}} E_{j,i}^{\beta_i} \stackrel{?}{=} \prod_{\ell < T} \Delta_{j,\ell}^{\sum_i \beta_i i^\ell}, \qquad \text{where}$$

$$\beta_i = H_{\mathsf{E}}\big(\mathsf{E\text{-}BATCH} \,\|\, e \,\|\, e{+}1 \,\|\, \nu \,\|\, j \,\|\, H(\mathbf{C}_j^{(e)}) \,\|\, H(\mathbf{C}_j^{(e+1)}) \,\|\, \{(i, E_{j,i})\}_i \,\|\, i\big),$$

with $H_{\mathsf{E}} : \{0,1\}^* \to \mathbb{Z}_q$ a second random oracle instantiated like H_{batch} and H a collision-resistant hash. This fixes every value the residuals depend on, including the two commitment vectors from which $\boldsymbol{\Delta}_j$ is derived, before any challenge is drawn, as Lemma 5 requires of the enrollment batch. A failed check identifies an inconsistent commitment family and, by binary search over the recipients, an invalid identifier i; the protocol below adjudicates (11) for that i. The contract does not accept $\Delta_{j,\ell}$ on trust from the dealer: the final step recomputes $\Delta_{j,\ell^\star}$ from the anchored per-dealer commitment vectors of the two states rather than from a separately signed delta vector.

Trace and endpoint authentication. Define $V_0 = 1_{\mathbb{G}}$ and $V_{\ell+1} = V_\ell \cdot \Delta_{j,\ell}^{i^\ell}$, so (11) holds exactly when $V_T = E_{j,i}$. The commitment vectors, the anchors of the two states and $E_{j,i}$ are fixed at finalization and authenticated by the dealer's signature and the record-membership proof of Appendix D.3, so both parties dispute a function of committed data. Each party commits to a Merkle root over the leaves (ℓ, V_ℓ) and must open both endpoints against its own root before anything else is compared: the challenger submits $\mathsf{Open}(r_C, 0, 1_{\mathbb{G}})$ and $\mathsf{Open}(r_C, T, z_C)$ with $z_C \neq E_{j,i}$, the dealer submits $\mathsf{Open}(r_D, 0, 1_{\mathbb{G}})$ and $\mathsf{Open}(r_D, T, E_{j,i})$, and a party that cannot produce these openings loses. No rule decides the dispute by comparing r_C with r_D: such a rule would let a dealer copy an honest challenger's root while claiming a different endpoint and prevail without violating any cryptographic binding, whereas with authenticated endpoints $r_C = r_D$ together with $z_C \neq E_{j,i}$ is itself a Merkle-binding violation and is treated as one.

Bisection. The contract maintains $[\mathsf{lo}, \mathsf{hi}] = [0, T]$ with the invariant that the authenticated values agree at lo and differ at hi, which the endpoint openings establish. A round is two transactions: at $m = \lfloor (\mathsf{lo} + \mathsf{hi})/2 \rfloor$ each party in turn submits $\mathsf{Open}(r, m, V_m)$, and the interval becomes $[m, \mathsf{hi}]$ if the two opened values agree and $[\mathsf{lo}, m]$ otherwise. A party that does not move within T_{round} loses by default, a verdict recorded separately from losing on a computation. After $\lceil \log_2 T \rceil$ rounds $\mathsf{hi} = \mathsf{lo} + 1 =: \ell^\star + 1$, and the transcript comprises $2\lceil \log_2 T \rceil$ openings of depth $\lceil \log_2(T+1) \rceil$, so the authentication data is $O(\log^2 T)$ hashes even though the round count is logarithmic.

Final step. Both parties have opened $V_{\ell^\star}$ and $V_{\ell^\star+1}$. The contract retrieves $C_{j,\ell^\star}^{(\sigma)}$ and $C_{j,\ell^\star}^{(\sigma')}$ by membership proofs against the two anchored commitment records, forms $\Delta_{j,\ell^\star}$, computes $i^{\ell^\star} \bmod q$, and compares $V_{\ell^\star} \cdot \Delta_{j,\ell^\star}^{i^{\ell^\star}}$ with $V_{\ell^\star+1}$. A party whose opened pair disagrees loses; if both disagree, both lose their deposits and no dealer is credited with having defended the record. The step costs at most two membership proofs, one per state and only for indices below that state's threshold, one inversion, one scalar exponentiation, one group exponentiation and one multiplication. The structure of narrowing a disputed computation to a single step is the interactive verification protocol of [33,49]; what is specific here is the relation being adjudicated and the fact that its inputs are recovered from anchored commitment vectors.

Theorem 7 (Consistency adjudication). *Assume the Merkle commitment is binding, the record-membership proofs are sound, the anchors are binding and the contract executes correctly. Then:* (completeness) *if* (11) *fails and the challenger follows the protocol within every deadline, the dealer loses, on the isolated step or by default;* (non-frameability) *if* (11) *holds and the dealer follows the protocol within every deadline, no challenger causes the dealer to lose on a computation; and* (timeout accountability) *a verdict by default is recorded as non-response and is distinguishable on chain from a verdict on the isolated step. The cost is* $G_{\mathsf{adm}} + G_{\mathsf{trace}} + 2\lceil \log_2 T \rceil G_{\mathsf{rnd}} + G_{\mathsf{fin}}$.

Proof. Endpoint authentication gives $V_0^C = V_0^D = 1_{\mathbb{G}}$ and $V_T^C = z_C \neq E_{j,i} = V_T^D$, so the interval invariant holds initially; each round preserves it and halves the interval, so after $\lceil \log_2 T \rceil$ rounds it isolates $\ell^\star$ with the opened values agreeing at $\ell^\star$ and differing at $\ell^\star + 1$. Merkle binding prevents either party from opening a value it did not commit.

Completeness. If (11) fails and the challenger commits to the true trace, which it computes from the anchored commitment vectors, its opened values satisfy every step relation and its endpoint $z_C = \prod_{\ell<T} \Delta_{j,\ell}^{i^\ell} \neq E_{j,i}$ opens correctly. The dealer must open T to $E_{j,i}$, so its committed sequence cannot both start at $1_{\mathbb{G}}$ and satisfy every step; the isolated pair it opens therefore fails the final check while the challenger's passes. If the dealer becomes unresponsive and misses the deadline, it loses by default instead.

Non-frameability. If (11) holds and the dealer commits to the true trace, the same argument with the roles exchanged shows that the challenger's isolated pair fails. The dealer's pair passes because the contract recomputes $\Delta_{j,\ell^\star}$ from the anchored commitment vectors rather than accepting a signed delta vector, so a challenger cannot win by disputing the delta vector itself.

Both parts assume one party follows the true trace; if neither does, both opened pairs may fail, which is why the rule makes both lose rather than crediting either. A default verdict carries a distinct status, so non-response is not conflated with a computation fault. □

F Extended Related Work

Table 8. Comparison with the closest refresh protocols. "Recipients passive" means that the parties whose issued shares are refreshed do not execute the refresh (in this work, the dealers hold coefficient shares and execute it, while participants only receive tokens); "later issuance" means that a new participant can receive a share of a committed sharing without modifying issued shares or the published verification state ("resharing" means that a new receiver is served only by a new sharing); "non-interactive recovery" means that a party absent throughout a refresh recovers without contacting the refreshing parties; and the last column records whether the security model treats an abandoned attempt as an object with its own budget, which is the criterion introduced in this paper; "n/a" marks a scheme without refresh. Entries reflect the stated scope of each work and do not claim that any work is insecure.

	Recipients passive	**Later issuance**	**Non-interactive recovery**	**Abandoned attempts budgeted**
PSS [32]	no	no	no	no
APSS, MPSS [55,46]	no	no	no	no
Dynamic committees [41,50,54]	no	no	no	no
Repairable threshold schemes [38]	n/a	yes	no	no
Offline refresh [37]	no	no	yes	no
Non-interactive resharing [29,27]	yes	resharing	yes	no
Independent refreshers [39]	yes	no	yes	not stated
This work	yes	yes	yes	yes

The last column concerns the security model, not a defect in the cited protocols: the question arises here because later issuance, a public delivery path for offline participants, and refresh attempts that may be abandoned after tokens are published occur together. For [39], the available description does not settle the question, so the entry reads "not stated".

Table 9 compares observable executions rather than names: what a corruption during a refresh yields, what an unfinished refresh leaves behind, whether a later key compromise re-exposes past material, and which feature of each model prevents its theorem from applying directly to our executions. The distinction we draw is narrow: it concerns the interface by which a party absent throughout a refresh later recovers from an archive alone, without contacting the refreshing parties, and the accounting of what an abandoned attempt leaves behind; it does not concern robustness to crashes, delays or Byzantine faults.

Table 9. Observable executions in the closest models, as we read their stated models. The comparison concerns model scope, not the security of the cited protocols in their own models.

Model	Corruption during a refresh	Left by an unfinished refresh	Later key compromise	Why the theorem does not transfer
PSS [32]	the party's shares and update messages, charged to both adjacent periods	no separate state: the update phase belongs to both periods	not applicable: private channels	holders run the refresh; no issuance to new parties and no archive-only recovery interface
Dynamic committees [41,54]	old- and new-committee state, charged to both committees	no separate state for an abandoned attempt; robustness under crash and Byzantine faults, asynchronously in [54]	not applicable: private channels	no interface for a party absent from the whole handoff to recover later from an archive alone
Non-interactive resharing [29]	the decrypted shares of corrupted receivers	published dealings persist	past ciphertexts stay protected by forward-secure encryption	receivers get a fresh sharing; no later issuance into a committed sharing
This work	source share, candidate partials or staged tokens, charged to source and candidate	tokens released to public recipients persist in the anchored log	archived records become readable, hence the public-recipient charge	—

The broader literature. Proactive secret sharing refreshes shares of an unchanged secret so that sub-threshold compromises do not accumulate [32], and has been extended to asynchronous networks [11,55], to mobile committees [46], to communication-optimal constructions for dynamic groups [2], and to committees with changing membership [41,50]; storing secrets on a public ledger for evolving committees is studied in [6,28], and role-based proactive sharing with user revocation in [31]. Our ReShare is a verifiable secret redistribution in the sense

of [15,53], specialized to a fixed committee and a fixed dealer threshold. Publicly verifiable secret sharing [48,45,12] addresses a related problem, but the asynchrony of asynchronous verifiable secret sharing [11] concerns agreement among the parties executing a sharing protocol, not a participant absent throughout a transition. Narrowing a disputed computation by bisection is the interactive verification game of [33,49]; ours is the relation adjudicated and the recovery of its inputs from anchored commitment vectors. KZG commitments are constant size but need a structured setup [34]; we use a transparent coefficient-wise Pedersen vector [42]. General secure multiparty computation [5] would suffice for dealer-side state evolution but is not required, because the generation layer is linear and its active security follows from Pedersen verifiable sharing and reliable broadcast in the manner of joint sharing protocols [43,25,26,1]. Evolving secret sharing [36,22] studies general access structures over an unbounded party sequence; we use later enrollment for the weaker property that within a committed epoch an arbitrary number of authorized participants may receive shares without modifying issued shares or the published verification state. Robust asynchronous dynamic-committee proactive secret sharing [54] and publicly verifiable secret sharing for thousands of parties [27] address the committee side of long-lived sharing. Accountable threshold signatures with proactive refresh [8] use accountability to identify signing quorums, whereas our dispute path attributes faulty update records and unmet publication obligations.

G Evaluation Details and Data

G.1 Component Measurements

System parameters and component environment. The component measurements use an Intel Core i9-14900K under Ubuntu 24.04, with $(n_D, k_D, f_D) = (4, 2, 1)$, $(7, 3, 2)$ and $(10, 4, 3)$. The component timings and analytical sizes assume a prime-order group whose elements compress to 32 bytes at $\lambda = 128$, with the commitment, the multi-recipient encryption and the signature scheme over that one group, and an execution layer supporting EIP-4844 blob transactions [9]. Table 10 distinguishes development chain transaction receipts from modeled gas costs.

Timings in Table 11 are measurements of the component benchmark; they serve as supplementary data to provide fine-grained component-level context. Record sizes and blob counts (Table 12) and the entries of Table 10 that are not transaction receipts are derived from the cost model rather than measured.

Admissible offline workloads. The source guard in (2) bounds the union of the incoming offline set, the release ledger and the outgoing set. For recipient sets of size o, one charged set gives $\delta + o < t$; disjoint incoming and outgoing sets give $\delta + 2o < t$; and A additional, disjoint sets from aborted attempts give $\delta + (A + 2)o < t$. At $t = 1024$ and $\delta = 32$, the first two bounds admit $o \leq 991$ and $o \leq 495$; the third admits $o \leq 330$ when $A = 1$. These are policy bounds, not empirical availability measurements. A recipient set rejected by the source guard

Table 10. On-chain cost. Receipts are from executed transactions on a local development chain; modeled figures are derived from the same contract with the stated operation counts. Calldata figures are given under the floor rule of [52].

Operation	**Gas**	**Source**
Contract deployment	2,521,346	receipt
Epoch initialization, 8 commitments	455,398	receipt
Collateral deposit	67,835	receipt
Blob-anchored epoch update (execution)	552,003	receipt
Blob-anchored epoch update (blob gas)	131,072	receipt
Complaint: signature and equality proof	30,302	receipt
Availability challenge: open	199,070	receipt
Availability challenge: finalize, no response	121,670	receipt
Anchor opening (point-evaluation precompile)	93,347	model
Calldata response, Merkle depth 1 / 3	32,878 / 34,598	model
Plaintext complaint against $E_{j,i}$	81,700	model
Plaintext complaint from $\Delta_{j,\cdot}$, $t = 64$	487,300	model
Plaintext complaint from $\Delta_{j,\cdot}$, $t = 1024$	6,390,000	model
Commitment vector in calldata, 32 KiB, floor rule	1,331,720	model

Table 11. Participant-facing costs at $t = 1024$ (medians; 50 trials for issuance and verification, 10 for the rest). The recovery row measures the delivery component *without* the proof-of-possession check of the final record format. The epoch commitment vector is 32 KiB at every enrolled population, so the marginal verification-state write per enrollment is zero.

Metric (ms)	2-of-4	3-of-7	4-of-10
Issuance	11.128	12.813	14.441
Share verification	7.676	7.683	7.675
Update-token application	9.530	10.315	11.105
Offline recovery (no PoP check)	40.947	53.514	62.496
Complaint generation / verification	0.265 / 0.330	0.269 / 0.332	0.268 / 0.332

remains ineligible for that source state, whose threshold is fixed and whose ledger is append-only. An attempt over already charged recipients does not further constrain this guard, so progress from the source state is not stalled; the attempt must still pass the target pre-check, generation and finalization (Appendix B.5).

Analytical prediction of the source reservation rule. The reservation rule is a test on the ledger, so its behavior is fixed by (2) rather than by measurement. At $t_\sigma = 8$, $\delta = 1$, $\mathcal{O}_\sigma = \{1, 2\}$ and an initially empty ledger, requesting $\{3, 4\}$ gives $1 + 4 < 8$ and reserves; $\{5, 6\}$ gives $1 + 6 < 8$ and reserves; $\{7\}$ gives $1 + 7 \not< 8$ and is refused with the ledger unchanged; and $\{3, 4\}$ again gives $1 + 6 < 8$ and is admitted, since those recipients are already charged. Appendix G.5 runs this sequence, together with a crash between reservation and release, on the reference model; the Rust implementation has not yet been tested against it.

Measurement coverage. The recovery timing in Table 11 excludes the proof-of-possession check required by (10), one equality-of-discrete-logarithms verification

per stored record, and measures only the aggregate phase of Section 3. The localization phase, with its $O(T)$ test and $2T$ group elements per candidate, is not measured; an adversary that malforms even a single selected record triggers the localization phase, so the cost $C_{\mathsf{recover}} = C_{\mathsf{agg}} + \mathbf{1}[(7)\text{ fails}]\, C_{\mathsf{loc}}$ has two branches that require separate measurement, with and without a cached commitment vector. Two cases of Lemma 2 are also not exercised. All component configurations satisfy (P2) with equality and have margin $m = 0$: they tolerate up to f_D malformed records but no additional loss of valid ones, whereas $(n_D, k_D, f_D) = (5, 2, 1)$ has $m = 1$ and would still leave k_D valid partials after one malformed and one unavailable record. And when two selected dealers shift their partials by the same u at coordinates whose Lagrange weights at 0 cancel, the aggregate phase accepts a correct token although both records violate (11). Localization, the release ledger and its crash and replay are exercised only on the reference model of Appendix G.5. The plaintext verifier, record admission, the consistency adjudication of Appendix E.4, which requires $2\lceil \log_2 T \rceil$ transactions (twenty at $t = 1024$, with 7,040 bytes of midpoint Merkle paths), the ordered reservation log, interactive re-issuance and the target-partial format are not measured. The per-coefficient authentication and source-vector retention of Appendix D.3 add publication costs that the epoch-update figures of Table 10 exclude. At $t = 16$, a separate test of the publication-admission condition (8) accepted record sets with f_D records removed and rejected them with $f_D + 1$ removed; this tests the publication quorum, not the weaker recovery condition of Lemma 2.

Generation layer. A transition consists of $n_D t_{e+1}$ Pedersen verifiable sharings of degree $k_D - 1$. Each dealer produces $t_{e+1} k_D$ commitments and $n_D t_{e+1}$ share pairs and verifies $n_D t_{e+1}$ sharings, each verification being one k_D-base multi-exponentiation. With 32-byte compressed group elements and 64-byte share pairs, the per-transition broadcast volume is $32\, n_D t_{e+1} k_D$ bytes and the point-to-point volume $64\, n_D^2 t_{e+1}$ bytes; at $t = 1024$ these are 256 KiB, 672 KiB and 1.25 MiB of broadcast and 1.0, 3.1 and 6.3 MiB of point-to-point traffic. None of these depends on the multiplicative complexity of a circuit, since no multiplication is performed. These communication volumes are analytical estimates; Table 15 reports generation times measured in the local lifecycle experiments. Measurements with wide-area network delay and public-chain consensus finality remain future work.

Participant-facing costs. At $t = 1024$, issuance takes 11.13–14.44 ms across the three configurations, with k_D partials checked in one multi-exponentiation, and share verification is independent of n_D (Table 11). At $t = 256$, issuance stayed within 4.14–4.68, 4.78–5.65 and 5.36–6.65 ms for the three configurations as the enrolled population grew from 1 to 2,048, and verification within 2.72–3.05 ms, with an 8,192-byte epoch commitment vector throughout; the marginal on-chain write of verification state per enrollment is zero. If (7) fails, localization retrieves the dealer's two commitment vectors, that is, $2T$ group elements or 64 KiB per

uncached candidate at $T = 1024$; caching removes that transfer but not the verification work or the fetches of replacement records.

Record size and storage. A signed record consists of the associated data, the common component (v, π^{epk}), the individual ciphertext, and the dealer signature. With the encryption instance of Appendix D.2 at $\lambda = 128$, these occupy 123, 128, 64 and 64 bytes, so a record is 379 bytes, of which 96 are the proof of possession and 32 the evaluation commitment; the one-byte mode field of (10) is included, and since $\lceil 378/31 \rceil = \lceil 379/31 \rceil = 13$ it does not change the padded size. Padded to 31-byte field elements, a record occupies 403 bytes, so a blob with 126,945 usable payload bytes, the first field element being reserved for the record root, holds exactly 315 records. With the publication quorum $n_D - f_D$, 2,048 offline participants require 20, 33 and 46 blobs (Table 12). Under the parameters in effect since the second blob-parameter-only fork, a target of 14 and a maximum of 21 blobs per block [40,17], this is one to three blocks at the maximum and two to four at the target. These figures use the modeled encoding. The implemented BN254 encoding is larger: comparing the transitions to epoch 1 with one and with sixteen returning participants in Table 17 gives about 857 additional payload bytes per record, or about 148 records per blob of the same usable size. The admissibility bounds of (2) require a correspondingly larger threshold for a large offline population.

Table 12. Blobs required by the public update log under the modeled encoding, at 379 bytes per record, 403 after field padding, and publication quorum $n_D - f_D$. These are capacity computations; the workload limits in Appendix G.1 state which of them correspond to admissible executions at a given threshold.

Offline	2-of-4	3-of-7	4-of-10
100	1	2	3
500	5	8	12
1000	10	16	23
2048	20	33	46

On-chain and storage cost. Table 10 separates receipts from models. Epoch-state publication grows linearly in the threshold, so a $t = 1024$ vector is published through blob-carrying transactions or calldata rather than contract storage. The primary architectural comparison lies between 81,700 gas for checking a decrypted partial against the per-recipient evaluation commitment and 487,300 gas at $t = 64$ (or 6.39 million gas at $t = 1024$) for recomputing the commitment from the delta vector, the latter rendering disputes cost-prohibitive at the thresholds targeted here. An evaluation-commitment dispute is a bisection protocol of $\lceil \log_2 T \rceil$ rounds, so at $t = 1024$ it requires ten on-chain rounds rather than a single transaction, incurring logarithmic rather than constant overhead. Calldata publication places the vector in execution-layer history, which is the option to choose when shares must be verified beyond the retention window; under

the floor rule of [52] it costs 1,331,720 gas for the 32 KiB vector, against 552,003 execution gas plus 131,072 blob gas for the blob route, so the ordering depends on the blob base fee and we report both rather than converting to currency.

G.2 End-to-End Workloads

All nine lifecycles run our Rust implementation from initial issuance to final reconstruction across epochs 0–3, under three workloads and three committee configurations $(n_D, k_D, f_D) = (4, 2, 1)$, $(7, 3, 2)$ and $(13, 5, 4)$, each fixed throughout a lifecycle (Table 13). Each lifecycle enrolls 256 new participants into a service of 1,024; the new participants enroll in epoch 1 at threshold 128, before the transition to epoch 2. The same o participants are offline for the transitions $0 \to 1$ and $1 \to 2$ and recover both updates after epoch 2 is finalized. Dealer authentication keys rotate after the transition to epoch 1, so the returning participants recover across both population growth and key rotation. All dealers and both archive processes remain online, and no corruption or malformed record is injected; f_D is used in the protocol quorums, not as a count of injected faults. The workloads satisfy (2): the source-side union never exceeds 16, so the source bound is $1 + 16 = 17$, and the target pre-check for the first two transitions is $1 + 16 + 16 = 33$, below the thresholds 128 and 160. The third transition has no public recipient but retains the incoming set in its source-side accounting.

Table 13. Evaluated workloads. Thresholds are indexed by epoch; o is the number of participants that miss the transitions $0 \to 1$ and $1 \to 2$, and ρ^{out} is the outgoing public-recipient reservation. The participant count is 1,024 in epochs 0 and 1 and 1,280 in epochs 2 and 3. Each row is executed once for each committee configuration, giving nine lifecycles; all use $\delta = 1$ and inject no faults.

Workload	t_0	t_1	t_2	t_3	o	ρ^{out}
Fixed threshold	128	128	128	128	1	2
Threshold increase	128	128	160	160	1	2
Multiple returning participants	128	128	160	160	16	16

Setting and timing. All executions use an Intel Core i9-14900K host with 32 logical processors. The owner, dealers, participants and archives are independent processes; private communication uses mutually authenticated TLS over the loopback interface. The implementation instantiates the commitments and the record encryption over BN254. A local Ethereum development node (Anvil) executes enrollment, the generation bulletin board, release reservations, blob publication and finalization, with execution block gas limits disabled. The measurements therefore include local communication, contract execution and transaction inclusion, but not wide-area network delay, public-chain finality or throughput under a public gas limit. The *lifecycle interval* runs from protocol initialization to final reconstruction and includes participant startup and enrollment, liveness

classification, generation, delivery, publication, finalization, key rotation, recovery and five reconstruction checks; chain deployment and initial process preparation are excluded. The *refresh interval* spans generation through finalization and online application. The *epoch interval* additionally includes the preceding enrollment and liveness classification, subsequent key rotation or recovery, and reconstruction.

G.3 Lifecycle and Transition Costs

Table 14 reports the complete executions. Lifecycle time ranges from 688.191 to 715.480 s for $(4, 2, 1)$, from 1,001.582 to 1,003.065 s for $(7, 3, 2)$, and from 1,688.199 to 1,756.229 s for $(13, 5, 4)$. Initial generation and issuance, together with the enrollment of 256 participants, account for 85.2–88.6% of these intervals. Enrollment includes registration, collection and verification of dealer partials, and on-chain activation before share installation; activations are confirmed sequentially. With 13 dealers and 16 returning participants, epoch 2 takes 406.974 s, of which 316.716 s is enrollment and 77.570 s is refresh, while the subsequent refresh of the same 1,280 participants takes 79.554 s. The committee configurations change n_D, k_D and f_D together, so they describe the cost of these configurations rather than isolating the number of dealers.

Table 14. Complete lifecycle costs in seconds, one execution per workload and committee. Initial denotes generation and issuance to the initial 1,024 participants; growth denotes enrollment of 256 participants. Growth is contained in epoch 2, so these columns are not additive. Total is the directly measured lifecycle interval, including reconstruction after initialization, after each transition and at the end. Final denotes the last reconstruction; deployment and initial process preparation are excluded.

n_D	Initial (s)	Growth (s)	Epoch 1 (s)	Epoch 2 (s)	Epoch 3 (s)	Final (s)	Total (s)
Fixed threshold, one returning participant							
4	469.680	127.000	27.236	156.378	27.224	3.928	688.191
7	699.387	188.521	33.560	225.996	35.773	4.022	1,002.701
13	1,160.648	310.520	65.762	382.394	70.804	4.353	1,688.199
Increased threshold, one returning participant							
4	472.818	125.918	27.122	159.681	33.010	5.986	702.335
7	684.370	188.029	33.346	231.789	43.707	5.866	1,003.065
13	1,167.335	311.570	64.768	396.367	86.546	6.343	1,725.553
Increased threshold, 16 *returning participants*							
4	483.824	126.856	27.599	161.660	32.589	5.800	715.480
7	681.279	187.455	34.360	233.256	42.955	5.854	1,001.582
13	1,183.540	316.716	67.541	406.974	87.486	6.407	1,756.229

Table 15 breaks each refresh into stages. With an increased threshold and 16 returning participants, generation in the transition to epoch 3 takes 1.865, 5.272 and 30.886 s for 4, 7 and 13 dealers, and the complete refresh intervals are 26.293, 36.311 and 79.554 s, so generation becomes a larger part of the refresh as the

committee grows. Comparing the two workloads with one returning participant, the increased threshold lengthens refresh by 10.5–16.7% at epoch 2 and by 16.9–21.3% at epoch 3, generation by 15.8–23.1%, and reconstruction by 47.1–55.5%; for 13 dealers, the epoch-3 refresh grows from 64.859 to 78.681 s. The complete lifecycles of these two workloads differ by only 0.04–2.21%, because both initialize at threshold 128 and enroll the new participants before the threshold increases. These are comparisons of single executions.

Table 15. Transition costs in seconds for all nine lifecycles. e identifies the target epoch. Gen.: generation; reserve: reservation and certification; direct: direct delivery and staging; publish: archive and blob publication; apply: finalization and online application; recon.: secret reconstruction. Refresh is measured from generation through application, including coordination between stages; it excludes enrollment, liveness classification, key rotation, recovery and reconstruction. Each entry is a single observation.

n_D	e	Gen. (s)	Reserve (s)	Direct (s)	Publish (s)	Apply (s)	Refresh (s)	Recon. (s)
Fixed threshold, one returning participant								
4	1	1.434	0.098	10.192	3.292	6.403	21.422	3.693
4	2	1.564	0.114	12.776	0.527	8.211	23.196	3.931
4	3	1.571	0.119	12.654	0.496	7.975	22.818	3.757
7	1	4.162	0.176	12.698	2.674	7.505	27.219	4.057
7	2	4.397	0.161	15.849	1.058	9.603	31.080	3.987
7	3	4.389	0.164	15.992	0.996	9.291	30.836	3.951
13	1	24.097	0.257	18.797	4.353	11.159	58.677	4.271
13	2	24.174	0.330	23.575	2.796	13.548	64.428	4.236
13	3	25.014	0.285	23.758	2.628	13.167	64.859	4.255
Increased threshold, one returning participant								
4	1	1.468	0.109	9.904	3.227	6.445	21.156	3.843
4	2	1.811	0.119	14.243	0.564	8.895	25.636	5.862
4	3	1.865	0.131	14.919	0.585	9.173	26.677	5.841
7	1	4.158	0.172	12.780	2.639	7.505	27.264	3.791
7	2	5.313	0.180	18.537	1.160	10.266	35.461	5.864
7	3	5.405	0.195	19.084	1.248	10.996	36.933	5.986
13	1	22.824	0.273	18.798	4.330	11.458	57.689	4.207
13	2	29.620	0.297	27.186	3.137	14.936	75.182	6.341
13	3	29.923	0.308	29.408	3.387	15.647	78.681	6.415
Increased threshold, 16 *returning participants*								
4	1	1.450	0.097	9.764	3.563	6.350	21.230	3.817
4	2	1.832	0.113	14.285	0.901	8.699	25.836	5.866
4	3	1.865	0.125	14.806	0.582	8.912	26.293	5.801
7	1	4.172	0.140	12.379	3.344	7.446	27.484	3.923
7	2	5.289	0.191	18.283	1.852	10.178	35.799	6.227
7	3	5.272	0.162	19.047	1.211	10.611	36.311	5.853
13	1	23.236	0.268	18.554	5.536	11.658	59.259	4.200
13	2	30.094	0.328	27.527	4.383	15.230	77.570	6.637
13	3	30.886	0.347	29.126	3.361	15.827	79.554	6.471

G.4 Recovery and Publication Costs

All returning participants recover both missed transitions from the archives without requesting fresh update partials from the dealers. Each recovery admits au-

thenticated records, decrypts them, verifies the aggregate update and applies it in epoch order, reading exactly k_D valid records per missed transition; no localization or additional retrieval of dealer vectors is needed in these executions. Table 16 distinguishes three timing scopes: one missed update for one participant; the individual interval covering both updates and the final state check, excluding restart and waiting for a recovery slot; and the wall-clock interval until the entire returning group has completed, including process restart and coordination. Up to 16 participants recover concurrently, each applying its two updates sequentially. With 16 returning participants, the group completes in 0.606, 1.016 and 2.652 s, with median individual intervals of 0.308, 0.583 and 1.842 s; with one returning participant, the group intervals are 0.241, 0.431 and 1.271 s. Individual intervals overlap and cannot be summed.

Table 16. Recovery times in seconds after epoch 2 finalizes. Update 1 and update 2 recover transitions $0 \to 1$ and $1 \to 2$. For 16 participants, each individual column gives the median and [minimum, maximum] across participants within that execution, with concurrency 16. Individual total covers both updates and final state verification, excluding restart and scheduling wait; this separate interval was not recorded for the one-participant workloads (dash). Group completion includes restart and coordination until all returning participants complete. Each participant reads $k_D = 2, 3, 5$ records per missed transition for $n_D = 4, 7, 13$, respectively.

n_D	Update 1 (s)	Update 2 (s)	Individual total (s)	Group completion (s)
Fixed threshold, one returning participant				
4	0.123	0.103	–	0.249
7	0.190	0.203	–	0.406
13	0.595	0.586	–	1.207
Increased threshold, one returning participant				
4	0.101	0.127	–	0.241
7	0.192	0.227	–	0.431
13	0.574	0.674	–	1.271
Increased threshold, 16 *returning participants*				
4	0.140 [0.092, 0.204]	0.142 [0.103, 0.221]	0.308 [0.208, 0.354]	0.606
7	0.261 [0.202, 0.446]	0.270 [0.207, 0.374]	0.583 [0.444, 0.758]	1.016
13	0.844 [0.675, 1.376]	0.907 [0.703, 1.236]	1.842 [1.569, 2.316]	2.652

Public update records are published only for participants that miss the corresponding transition, one record per dealer and participant in these executions: n_D records for one offline participant and $16n_D$ for sixteen. The third transition publishes no recipient record but still anchors the authenticated state. Table 17 reports payloads, blob counts and publication gas. Payloads include coefficient information and metadata, so their size does not scale only with the number of returning participants: at epoch 3 with 13 dealers and no offline participant, the payload is 126,328 bytes at threshold 128 and 157,048 bytes at threshold 160,

which changes the blob count from one to two. Every measured publication fits in one transaction.

Table 17. Public update data and on-chain publication costs for each target epoch e. The record count includes only recipient update records; payload is the encoded bundle size in bytes, including authenticated coefficient information and metadata, before padding into blobs. Each row uses one publication transaction. Execution gas and blob gas are taken from its receipt and exclude other lifecycle transactions. The zero-record transitions still publish state information. Publication latency is reported in Table 15.

n_D	e	Records (count)	Payload (bytes)	Blobs (count)	Execution gas	Blob gas
Fixed threshold, one returning participant						
4	1	4	53,944	1	603,318	131,072
4	2	4	53,944	1	682,918	131,072
4	3	0	50,512	1	682,918	131,072
7	1	7	81,784	1	761,046	131,072
7	2	7	81,784	1	900,370	131,072
7	3	0	75,784	1	900,382	131,072
13	1	13	137,464	2	1,154,830	262,144
13	2	13	137,464	2	1,413,578	262,144
13	3	0	126,328	1	1,335,298	131,072
Increased threshold, one returning participant						
4	1	4	53,944	1	603,306	131,072
4	2	4	60,088	1	682,918	131,072
4	3	0	62,800	1	682,930	131,072
7	1	7	81,784	1	761,070	131,072
7	2	7	91,000	1	900,382	131,072
7	3	0	94,216	1	900,394	131,072
13	1	13	137,464	2	1,154,866	262,144
13	2	13	152,824	2	1,413,578	262,144
13	3	0	157,048	2	1,413,506	262,144
Increased threshold, 16 *returning participants*						
4	1	64	105,424	1	603,406	131,072
4	2	64	111,568	1	683,006	131,072
4	3	0	62,800	1	683,018	131,072
7	1	112	171,784	2	839,474	262,144
7	2	112	181,000	2	978,762	262,144
7	3	0	94,216	1	900,458	131,072
13	1	208	304,504	3	1,233,245	393,216
13	2	208	319,864	3	1,491,921	393,216
13	3	0	157,048	2	1,413,654	262,144

Scope. The lifecycle measurements characterize normal execution on a single host with fixed dealer membership and one execution per workload and committee. Differences in liveness classification, recovery scheduling and verification checks across the executions limit cross-workload comparisons to descriptive observations. Aborted attempts, disputes, long-term archive availability and wide-area performance are outside the lifecycle evaluation.

G.5 Executable Reference Model

The reference model is implemented in Python, independently of the Rust implementation. It has three layers. The *cryptographic layer* works in the 2048-bit prime-order group of [35] and provides hash-to-field in the style of [18], Schnorr signatures, Merkle anchors over record digests, and a record encryption with the interface of Section 3: an ephemeral key with a proof of possession bound to the associated data, a key derived from pk_i^{κ} and the associated data, and a Chaum–Pedersen decryption proof [13]. This encryption is a simplified instance with that interface, not a reimplementation of [30]; the traces therefore check functional behavior and do not validate the associated-data-only chosen-ciphertext assumption of Theorem 3 or an implementation of [30]. The *ledger layer* stores every reservation, release and abort in an append-only log that is flushed with `fsync` before the operation returns. ReserveRelease is a compare-and-set on a root that hashes the ledger version and contents: it evaluates the source condition of (2), requires a certificate of $n_D - f_D$ valid reservation signatures on $(\sigma, \mathsf{root}_{\mathsf{old}}, \nu, \mathcal{O}_a, \mathsf{root}_{\mathsf{new}})$, rejects a stale root, treats a repeated identical request as idempotent, and rejects the reuse of a nonce with different content. A release requires a durable reservation whose certificate verifies, and a restarted process rebuilds its state by replaying the log. The model has a single ledger replica, so it does not test the replication protocol of Appendix B.4. The *recovery layer* holds bivariate dealer states with Pedersen commitments and runs the aggregate and localization phases of Section 3. In the traces, admission verifies the dealer signature, the anchor membership proof and the proof of possession, and each plaintext is obtained by decryption.

Traces. Table 2 summarizes the scenarios. The ledger scenarios use $(n_D, f_D) = (4, 1)$, so a certificate needs three signatures. In the crash scenario, a child process performs the certified reservation and terminates with exit status 137 before any release; the restarted process finds the recipient in $\widehat{U}_\sigma$ and none released, which is the conservative direction of Lemma 1, and it can then release the attempt, whereas a release without a durable reservation is blocked. In the cancelling scenario, corrupted dealers 1 and 2 add offsets whose Lagrange weights cancel and sign matching evaluation commitments, so their records pass admission and opening; the aggregate phase accepts the correct token, and an audit against (11) shows that only dealer 3's record is valid. A record that claims an honest dealer's index but is signed with another key, and a record with an invalid proof of possession, fail admission. A record whose plaintext does not open $E_{j,i}$ is discarded; a complaint with a verified decryption proof charges its dealer, and a complaint against an honest record charges the claimant. With $(n_D, k_D, f_D) = (5, 2, 1)$ and $m = 1$, one malformed and one unavailable record still leave k_D valid ones; recovery ends in localization after 3 reads. After two missed transitions with $(7, 3, 2)$, difference records required the source share and 6 reads, whereas snapshot records required 3 reads and no source share; both returned the correct share.

Randomized check of Lemma 2. The randomized check exercises the recovery arithmetic; admission is represented by a flag, because the traces cover its cryptographic layer. Each trial draws source and target thresholds between 2 and 4, a random identifier, a random canonical order of the dealers, up to f_D faulty records (a wrong evaluation commitment, a failed opening, a failed admission, or a pair whose offsets cancel in the canonical combination), and up to m unavailable valid records. Every parameter set satisfies Assumption 1; targeted runs place a cancelling pair at the head of the canonical order in every trial. In every trial of Table 18, the returned share equals the target share and verifies against $\mathbf{C}^{(e+1)}(i)$, the number of reads is at most $k_D + f_D$, every cancelling pair is accepted by the aggregate phase, and every localization combines only records that satisfy (11).

Table 18. Randomized check of Lemma 2 on the reference model. "Cancel" counts trials in which a cancelling pair entered the canonical set and the aggregate phase accepted.

(n_D, k_D, f_D)	m	Seed	Trials	Aggregate	Localization	Cancel	Max reads	$k_D + f_D$
$(4, 2, 1)$	0	7042	30	28	2	0	3	3
$(5, 2, 1)$	1	7052	30	27	3	0	3	3
$(7, 3, 2)$	0	7073	30	25	5	0	5	5
$(8, 3, 2)$	1	7083	30	26	4	1	5	5
$(10, 4, 3)$	0	7104	20	14	6	0	7	7
$(7, 3, 2)$ targeted	0	8073	15	15	0	15	3	5
$(8, 3, 2)$ targeted	1	8083	15	15	0	15	3	5