

DASH: Distributed Asynchronous Schnorr with High Throughput

Jiawei Ni , Yanhong Xu, Chen Yuan and Hongqing Liu

Shanghai Jiao Tong University, Shanghai, China

Abstract. Robust asynchronous threshold Schnorr signatures enable honest parties to complete signing despite malicious behavior and network delays. To reduce online latency, these schemes typically follow an offline–online paradigm. The parties prepare batches of shared nonces in the offline phase and consume them to generate signatures online. In high-throughput constructions such as SPRINT (Benhamouda et al. Eurocrypt 2024), super-invertible matrices are used to extract multiple nonces and their corresponding commitments from a single execution of the nonce-generation protocol. However, the associated group operations incur substantial computational overhead.

We identify several computational bottlenecks in SPRINT and present DASH, a suite of robust asynchronous threshold Schnorr signature protocols that substantially improve computational efficiency while retaining SPRINT’s amortized linear communication cost. Our contributions are threefold. First, we propose distributing the group computations involved in nonce extraction among the parties. This reduces the amortized computational cost from $\mathcal{O}(\log n)$ to $\mathcal{O}(1)$ group operations per signature per party. To enable this distributed computation, we employ a batching technique that allows each dealer to distribute L nonce polynomials simultaneously. Second, we introduce an efficient batched share-verification procedure that jointly checks shares of all L polynomials. For concrete parameters, this can accelerate share verification by a factor of 40–60. We also introduce a simple agreement protocol via player elimination, which could be of independent interest. Finally, under a stronger honest-majority assumption, we develop a packed two-stage online reconstruction protocol that enables parties to reconstruct signatures through error correction. This eliminates the need for signature-share verification, and consequently, avoids the expensive group operations performed by the aggregator in SPRINT. Surprisingly, even in the worst case, our DASH protocols preserve the same asymptotic amortized computation and communication costs, provided that $L \gg n \log n$.

1 Introduction

Threshold cryptography [16,17] enables a set of n parties to jointly perform cryptographic operations using a shared secret key while tolerating the corruption of up to t parties. As a prominent application of threshold cryptography, threshold signatures enable n signers to collaboratively generate a valid signature on a

message, even in the presence of up to t corrupted parties. Threshold Schnorr signatures [31] have received growing attention for their use in blockchain systems and cryptocurrency custody [27]. Early work established foundations for threshold cryptography [13] and threshold Schnorr signing [34,1,20]. Subsequent research has improved signing efficiency and security [26,7,14], with further developments in asynchronous robustness and throughput [30,9,23,4] and adaptive security [3,15,2,21,32]. These schemes offer a wide range of trade-offs among efficiency, round complexity, security against static or adaptive adversaries, network synchrony assumptions, and robustness. In this work, we focus on designing efficient threshold Schnorr signature schemes with robustness in the asynchronous setting.

When deployed in Blockchain systems and cryptocurrencies, threshold signatures with robustness ensure the liveness of the system. This guarantees that the system will continue to function even if some parties crash, lose their key shares, or behave maliciously. However, protocols that provide security with abort, such as FROST [26], offer no such guarantee and may involve expensive manual intervention.

In the synchronous setting, all honest parties are assumed to finish each round within a fixed time bound. However, on the one hand, this bound may be set conservatively to account for network delays, causing high network latency. On the other hand, it may be set aggressively so that the security of the protocols may be violated. Asynchronous protocols avoid this dilemma without setting any time bound, and can proceed as fast as the network allows. Indeed, the combination of robustness and asynchrony in threshold protocols is particularly powerful, as the protocol can make progress without waiting for slow or malicious parties.

Regarding efficiency, providing robustness generally incurs additional communication and computational overhead and increases round complexity of threshold Schnorr signature schemes. e.g., [34,20,30,4,9,23,32]. The protocols proposed by Stinson and Stroh [34] and by Gennaro et al. [20] require at least four rounds to generate a signature (assuming no signers misbehave): three rounds to generate message-independent ephemeral randomness and one round to distribute signature shares. In addition, it requires broadcasting $\mathcal{O}(n^2)$ group elements for the randomness generation. This will then result in communication complexity $\mathcal{O}(\lambda n^3)$ bits broadcast ¹ per signature (λ is the security parameter). ROAST [30] presents a wrapper technique that turns a given non-robust threshold signature scheme into a scheme with robustness even in the asynchronous setting. However, achieving robustness in the worst case comes at a substantial cost: it requires $\mathcal{O}(tn^2 + tn\lambda)$ bits of communication per signature with the assistance of a trusted coordinator and $\mathcal{O}(tn^3 + tn^2\lambda)$ bits without the coordinator. Moreover, achieving worst-case robustness may require up to $\mathcal{O}(n)$ communication rounds.

¹ According to [18,28,33], reliably broadcasting an l -bit message, where $l \geq \lambda n \log n$, requires $\mathcal{O}(ln)$ bits of point-to-point communication.

Bacho et al. [4] proposed HARTS, a robust threshold Schnorr signature scheme for the asynchronous setting that aims to achieve adaptive security² while supporting a high reconstruction threshold. They employed the technique of super-invertible matrices [25] to extract ephemeral randomness to achieve an amortized communication cost of $\mathcal{O}(\lambda n^2 \log n)$ bits per signature. The computational cost of HARTS is quite high, as it relies on expensive group operations. Benhamouda et al. [9] presented a (statically secure and) robust threshold Schnorr signature SPRINT that operates in the asynchronous setting. Compared to HARTS, SPRINT achieves lower communication and computational complexity by combining packed Shamir secret sharing and super-invertible matrices to amortize the associated costs, at the expense of a weaker resilience bound $n \geq 3t + 2a - 1$. Here, a is the packing parameter. Specifically, SPRINT incurs $\mathcal{O}(\lambda n)$ bits of communication per signature and $\mathcal{O}(\log n)$ group operations per signature per party. It is worth noting that both HARTS and SPRINT generate commitments to $\mathcal{O}(n)$ evaluations of the nonce polynomials used for sharing randomness. These commitment operations, together with the subsequent randomness extraction, dominate the computational cost of both protocols.

To reduce the computational cost, Groth and Shoup [23] presented a robust threshold Schnorr signature scheme through a lightweight asynchronous verifiable secret sharing (AVSS) [33]. At the heart of their construction is a batching technique inherited from [33], a simple statistical test such that the work can be distributed among n parties, and also a novel construction of a super-invertible matrix for randomness extraction. The protocol has the same $\mathcal{O}(\lambda n)$ bits of communication as SPRINT if the batching size $L \gg n \log n$, and requires $\mathcal{O}(n)$ group additions. Nevertheless, the protocol [23] has the optimal resilience threshold $n \geq 3t + 1$. However, the GS24 protocol [23] incurs an amortized communication cost of $\mathcal{O}(\lambda n^2)$ per signature in the worst case, due to the complaint and share recovery procedures triggered by malicious dealers. Shoup [32] upgraded this protocol to achieve adaptive security with erasure. To avoid this quadratic communication complexity, Shoup presented a simplified protocol that achieves worst-case linear communication complexity at the cost of a weaker resilience bound $n \geq 4t + 1$. Importantly, they forgo the completeness property of the AVSS protocol and use a star-finding algorithm [12] to agree on which dealers and receivers to use for a given presignature batch. Note that completeness implies that all honest parties will eventually receive shares of the shared secret.

To the best of our knowledge, existing robust threshold Schnorr signature schemes that achieve $\mathcal{O}(\lambda n)$ bits of communication per signature in the asynchronous setting [9, 23, 32] all rely on super-invertible matrices for ephemeral randomness extraction. Such nonce extraction remains a computational bottleneck for the above three protocols. Particularly, each party independently performs this nonce extraction step to derive ephemeral randomness. Consequently, the computational cost per signature per party grows with the number of parties, typically requiring either $\mathcal{O}(n)$ or $\mathcal{O}(\log n)$. Therefore, it is encouraging to consider distributing the randomness extraction workload among the n parties,

² The authors acknowledge a gap in the adaptive security proof [5].

so that each party performs only a small number of group operations, possibly achieving $\mathcal{O}(1)$ computational complexity. This motivates the following question.

Question 1. Assuming a constant fraction of corrupted parties, can we construct a robust threshold Schnorr protocol in the asynchronous setting with linear communication per signature and constant computation per signature per party?

1.1 Our Contributions

In this work, we answer the above question affirmatively by introducing DASH, a suite of threshold Schnorr protocols that tolerate up to t corruptions among n parties. To this end, we integrate the packed Shamir secret sharing scheme with a packing parameter a , batching techniques with batching size L and a super-invertible matrix to extract randomness. Concretely, assuming $t = \Theta(n)$ and $a = \Theta(n)$, we present three variants of DASH with the following properties. We refer to Table 1 for a complete overview of our protocols and comparison with some prior works.

- **Asynchrony:** All three variants of DASH are secure in the asynchronous setting. To withstand t corrupted parties, DASH1 and DASH2 require $n \geq 3t + 2a - 1$, and DASH3 requires $n \geq 4t + 2a - 1$.
- **Linear communication:** In the optimistic case, where no party misbehaves, all variants of DASH incur a total amortized communication cost of $\mathcal{O}(\lambda n)$ bits per signature, provided that $L \gg n$.
- **Constant computation:** In both optimistic and pessimistic cases, all variants of DASH incur an amortized computational cost of $\mathcal{O}(1)$ group operations per signature per party, provided that $L \gg n$.
- **Robustness:** All variants of DASH are robust and the additional communication overhead to achieve it is moderate. When parties misbehave, the protocols invoke an additional complaint mechanism. Specifically, DASH1 incurs a total communication of $\mathcal{O}(\lambda n^3 \log n)$ bits while DASH2 and DASH3 introduce $\mathcal{O}(\lambda n^4 \log n)$ bits for a total of $\Theta(n^2 L)$ signatures. Consequently, the complaint mechanism contributes only $\mathcal{O}(\lambda n \log n / L)$ amortized communication per signature for DASH1, and $\mathcal{O}(\lambda n^2 \log n / L)$ for DASH2 and DASH3. Hence, this additional robustness overhead does not change the asymptotic amortized communication cost of $\mathcal{O}(\lambda n)$ bits per signature, provided that $L \gg \log n$ for DASH1 and $L \gg n \log n$ for DASH2 and DASH3.

1.2 Technical Overview

Overview of SPRINT. Our starting point is SPRINT [9], a robust threshold Schnorr signature protocol that combines packed secret sharing with batched randomness extraction in an asynchronous network. Let $(\mathbb{G}, +)$ be a group of prime order p with a generator G . Recall that a Schnorr signature on a message M consists of (R, σ) , where $R = r \cdot G$ is the partial signature and $\sigma = r + e \cdot s$ for the signing key $s \in \mathbb{F}_p$ and challenge $e = \text{hash}(\text{vk}, R, M)$, with $\text{vk} = s \cdot G$.

Table 1. Comparison of some robust asynchronous threshold Schnorr protocols in the optimistic case. Min. n denotes the minimum number of parties required to tolerate t corruptions. Communication measures total transmitted bits per signature, and computation measures group operations per signature per party.

Scheme	Min. n	Offline		Online	
		Comm.	Comp.	Comm.	Comp.
ROAST [30]	$2t + 1$	$\mathcal{O}(n\lambda)$	$\mathcal{O}(1)$	$\mathcal{O}(n^2 + n\lambda)$	$\mathcal{O}(n)$
HARTS [4]	$3t + 1$	$\mathcal{O}(n^2\lambda \log n)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2\lambda)$	$\mathcal{O}(n)$
SPRINT [9]	$3t + 2a - 1$	$\mathcal{O}(n\lambda)$	$\mathcal{O}(\log n)$	$\mathcal{O}(n\lambda)$	$\mathcal{O}(1)$
GS24 [23]	$3t + 1$	$\mathcal{O}(n\lambda)$	$\mathcal{O}(n)$	$\mathcal{O}(n\lambda)$	$\mathcal{O}(1)$
AHAB [32]	$4t + 1$	$\mathcal{O}(n\lambda)$	$\mathcal{O}(n)$	$\mathcal{O}(n\lambda)$	$\mathcal{O}(1)$
DASH	$3t + 2a - 1^*$	$\mathcal{O}(n\lambda)$	$\mathcal{O}(1)$	$\mathcal{O}(n\lambda)$	$\mathcal{O}(1)$

* Our lightweight variant DASH3 requires $n \geq 4t + 2a - 1$.

For n parties with at most t corruptions, SPRINT uses a packing factor a and requires $n \geq 3t + 2a - 1$. The long-term key is shared through a degree- d polynomial $\mathbf{F}$, where $d = t + a - 1$ and $\mathbf{F}(1 - v) = s$ for every $v \in [a]$. Thus, each party P_i holds a share $[\mathbf{s}]_i = \mathbf{F}(i)$ of $\mathbf{s} = (s, \dots, s) \in \mathbb{F}_p^a$, together with the publicly known verification key share $\mathbf{vk}_i = \mathbf{F}(i) \cdot G$. This packing allows one share to contribute to a signatures under the same signing key.

For signature generation, SPRINT follows an offline-online paradigm: the parties first prepare shared nonces and the corresponding partial signatures, then use these values to generate signatures once the messages are known.

During the offline phase, each party P_i acts as a dealer and shares a random degree- d' nonce polynomial $\mathbf{H}_i$, where $d' = t + 2a - 2$. Concretely, the dealer broadcasts commitments $\hat{\mathbf{H}}_i(v) = \mathbf{H}_i(v) \cdot G$ for $v \in [1 - a, d' - a + 1]$ and ciphertexts of $s_{i,j} = \mathbf{H}_i(j)$ for $j \in [n]$. Each shareholder P_j decrypts the share $s_{i,j}$ and checks whether $s_{i,j} \cdot G \stackrel{?}{=} \sum_{v=1-a}^{d'-a+1} \lambda_{i,j,v} \hat{\mathbf{H}}_i(v)$, where $\lambda_{i,j,v}$ are the relevant Lagrange coefficients. If the check fails, P_j broadcasts a publicly verifiable complaint containing $s_{i,j}$ and a proof of correct decryption.

A main contribution of the SPRINT protocol is a simple agreement protocol. The protocol uses these complaints to select a set **QUAL** of qualified dealers and a set **HOLD** of receivers, referred to as shareholders in SPRINT. The essential guarantee is that every honest shareholder in **HOLD** holds valid shares from every dealer in **QUAL**, with enough honest shareholders to reconstruct the signatures. This avoids requiring every honest party to obtain every retained sharing, which is unnecessary for signing and costly to ensure in an asynchronous network.

To obtain many nonces from the dealers' contributions, SPRINT utilizes a public super-invertible matrix $\Phi \in \mathbb{F}_p^{b \times c}$, where $c = |\mathbf{QUAL}|$ and $b = n - 2t$ is a lower bound on the number of honest qualified dealers. Let $\mathbf{QUAL} = (P_{Q_1}, \dots, P_{Q_c})$. Define a matrix $X \in \mathbb{G}^{c \times a}$, where the entry in the j -th row

and v -th column is $X_{j,v} = \hat{\mathbf{H}}_{Q_j}(1-v)$. The partial signatures are then given by $(R^{(u,v)})_{u \in [b], v \in [a]} = \Phi X$. Thus, a single execution of the nonce-generation protocol yields ab partial signatures.

Let $(\mathbf{H}^{(1)}, \dots, \mathbf{H}^{(b)})^\top = \Phi \cdot (\mathbf{H}_{Q_1}, \dots, \mathbf{H}_{Q_c})^\top$. In the online phase, on input messages $M^{(u,v)}$ for $u \in [b], v \in [a]$, each shareholder computes $e^{(u,v)} = \text{hash}(\text{vk}, R^{(u,v)}, M^{(u,v)})$. For $u \in [b]$, let $\mathbf{Z}^{(u)}$ be the unique polynomial of degree at most $(a-1)$ satisfying $\mathbf{Z}^{(u)}(1-v) = e^{(u,v)}$ for $v \in [a]$. Each shareholder P_i in HOLD then locally computes its shares of polynomials $\mathbf{Y}^{(u)} := \mathbf{H}^{(u)} + \mathbf{Z}^{(u)}\mathbf{F}$ for $u \in [b]$. It is worth mentioning that one can verify the validity of those shares by checking whether $\mathbf{Y}^{(u)}(i) \cdot G = \mathbf{H}^{(u)}(i) \cdot G + \mathbf{Z}^{(u)}(i) \cdot \text{vk}_i$ holds. In SPRINT, $|\text{HOLD}| = n - t$, and hence there are at least $n - 2t \geq t + 2a - 1 = d' + 1$ valid shares. As a result, one can recover the degree- d' polynomial $\mathbf{Y}^{(u)}$ and hence signatures on $M^{(u,1)}, \dots, M^{(u,a)}$.

By combining packed secret sharing with super-invertible matrices, SPRINT amortizes the communication cost over the ab signatures, achieving $\mathcal{O}(n\lambda)$ bits per signature when $a = \Theta(n)$. However, its computational cost remains high because of the group operations required for (i) computing the partial signatures $\{R^{(u,v)}\}_{u \in [b], v \in [a]}$ during the offline phase, (ii) verifying the shares distributed by dealers before sending the complaint messages, and (iii) computing $\mathbf{H}^{(u)}(i) \cdot G$ to verify the signature shares $\mathbf{Y}^{(u)}(i)$ during the online phase.

In this work, we progressively address these computational bottlenecks, yielding three protocol variants: DASH1, DASH2, and DASH3.

Distributed partial signature generation. In SPRINT, every party independently computes the same matrix product $(R^{(u,v)})_{u \in [b], v \in [a]} = \Phi X$. The authors employ ECFFT-EXTEND to reduce the cost of this step to $\mathcal{O}(n^2 \log n)$ group operations. Amortized over the resulting ab signatures, this yields a computational cost of $\mathcal{O}(\log n)$ group operations per signature per party.

To achieve $\mathcal{O}(1)$ amortized computational complexity, we instead distribute the workload among the parties. Each party performs only a fraction of the overall computation and sends the result to the others, who verify the received result rather than recomputing it themselves. To this end, we would like to partition X into n blocks $X = [X_1, \dots, X_n]$ such that each party P_i computes $\tilde{R}_i = \Phi \cdot X_i$. Upon receiving $\tilde{R}_j$ for $j \in [n] \setminus \{i\}$, P_i verifies the computation by checking whether $\lambda^{(i,j)} \cdot \tilde{R}_j = (\lambda^{(i,j)} \cdot \Phi) \cdot X_j$ holds for some random $\lambda^{(i,j)} \in \mathbb{F}_p^b$. This approach, however, raises two issues. First, X contains only $a < n$ columns and therefore cannot be partitioned into n non-empty column blocks as described above. Second, the randomized verification is not perfectly sound: even when the check succeeds, there remains a non-zero probability that $\tilde{R}_j$ was computed incorrectly.

To address the first issue, we employ a batching technique, inspired by [33,23,32]. Concretely, each dealer P_i now shares L nonce polynomials $\mathbf{H}_i^{(1)}, \dots, \mathbf{H}_i^{(L)}$ and the remaining steps of SPRINT are extended accordingly. In particular, the parties compute $(R^{(u,v,\ell)})_{u \in [b], v \in [a], \ell \in [L]} = \Phi \cdot X \in \mathbb{G}^{b \times aL}$ for some $X \in \mathbb{G}^{c \times aL}$. In our protocols, we choose $L \gg n \log n$ so that aL will be sufficiently large to partition X into n blocks. This resolves the first issue.

To address the second issue, we analyze the soundness error of the randomized verification and show that it is bounded by $(n - t)t/p$. Thus, for sufficiently large p , all honest parties agree on the same correctly computed values $\tilde{R}_j$ with overwhelming probability. This will then reduce the computational cost to $\mathcal{O}(1)$ per signature per party. SPRINT augmented with the above modifications gives our base construction, DASH1.

Batched share verification with player elimination. In DASH1, checking L nonce shares from each of the c qualified dealers requires $\mathcal{O}(|\text{QUAL}|d'L)$ full scalar-group multiplications. Concretely, P_j has to check $s_{i,j}^{(\ell)} \cdot G = \sum_{v=1-a}^{d'-a+1} \lambda_{i,j,v} (\mathbf{H}_i^{(\ell)}(v) \cdot G)$, for $P_i \in \text{QUAL}$, and $\ell \in [L]$. Instead of checking these L polynomials individually, we check them together using a batched verification technique. We choose random values $\gamma_i^{(\ell,k)}$ for $\ell \in [L], k \in [K]$ for some integer $K \leq n$ after the dealer P_i has finished its broadcast step. Then P_i replies with K response polynomials $\{\mathbf{C}_i^{(k)} = \sum_{\ell \in [L]} \gamma_i^{(\ell,k)} \cdot \mathbf{H}_i^{(\ell)}\}_{k \in [K]}$. Then P_j performs the following checks: for each $P_i \in \text{QUAL}$,

- For each $k \in [K]$, check whether $\deg(\mathbf{C}_i^{(k)}) \leq d'$ and $\mathbf{C}_i^{(k)}(j) = \sum_{\ell \in [L]} \gamma_i^{(\ell,k)} s_{i,j}^{(\ell)}$ hold. This step requires KL field operations.
- For every $v \in [1 - a, d' - a + 1]$, check whether $\mathbf{C}_i^{(m_j)}(v) \cdot G = \sum_{\ell \in [L]} \gamma_i^{(\ell,m_j)} \cdot (\mathbf{H}_i^{(\ell)}(v) \cdot G)$, where $m_j = (j - 1) \bmod K + 1$. This step requires $d'L$ scalar-group multiplications.

At first glance, the above approach appears to require $|\text{QUAL}| \cdot d'L$ scalar-group multiplications, together with additional field operations and communication. However, there is an important difference in the size of the scalars involved. While the scalars $\lambda_{i,j,v}$ used in DASH1 are drawn from $\mathbb{F}_p$, the random coefficients $\gamma_i^{(\ell,k)}$ can be sampled from a much smaller subset $\Gamma \subseteq \mathbb{F}_p$. Consequently, the corresponding scalar-group multiplications are substantially cheaper. For concrete parameters, share verification is approximately 40 – 60 times faster. We further prove that, except with negligible probability, all shares distributed by a dealer are valid whenever the above batch checks succeed. To preserve privacy, we additionally introduce K blinding polynomials $\{\mathbf{G}_i^k\}_{k \in [K]}$.

The above mechanism allows each party to determine whether the shares received from a dealer are valid. It remains to design an agreement protocol through which the parties agree on the sets **QUAL**, **HOLD**. A straightforward method would be to follow the agreement mechanism of SPRINT: when P_j complains against P_i , it reveals $s_{i,j}^{(\ell)}$ for all $\ell \in [L]$ together with proofs of correct decryption. It is worth noting that revealing all L values is necessary because the batch verification only detects that some share is invalid, without identifying which one. However, the additional cost of such a complaint mechanism would offset the efficiency gains achieved by batch verification. Instead, we allow P_j to complain against P_i without providing any evidence, and eliminate both parties from the active party set. We then carefully design a new agreement protocol such that all honest shareholders agree on **QUAL** and **HOLD**, while guaranteeing

that every honest shareholder in **HOLD** receives valid shares from every dealer in **QUAL**. Incorporating these modifications into **DASH1** yields our **DASH2** protocol.

Two-stage online reconstruction with packing. During the online phase, **SPRINT** resorts to an aggregator to perform the expensive computation of $\{\mathbf{H}^{(u)}(i)\}_{u \in [b], P_i \in \text{HOLD}}$. A natural question is whether the aggregator can be eliminated and the shareholders can instead reconstruct the signatures by themselves. The answer is affirmative under a stronger honest majority assumption. Recall that each $P_i \in \text{HOLD}$ computes $\mathbf{Y}^{(u)}(i) := \mathbf{F}(i) \cdot \mathbf{Z}^{(u)}(i) + \mathbf{H}^{(u)}(i)$ in **SPRINT**, where $\mathbf{Y}^{(u)}$ has degree $d' = t + 2a - 2$. Since $|\text{HOLD}| = n - t$, there are at least $n - 2t$ honest shareholders in **HOLD**. Hence, if $n - 2t \geq d' + 1 + t$, then the $n - t$ shares held by **HOLD**, among which at most t may be erroneous, uniquely determine the degree- d' polynomial $\mathbf{Y}^{(u)}$. Substituting $d' = t + 2a - 2$ gives $n \geq 4t + 2a - 1$. Hence, under this stronger resilience condition, the shareholders in **HOLD** can reconstruct the signatures without verifying the validity of shares $\mathbf{Y}^{(u)}(i)$. This then eliminates the assistance of an aggregator.

Concretely, the signature reconstruction proceeds in two stages. Let $\sigma_1, \dots, \sigma_B$ be B signatures that we need to reconstruct. Each $P_i \in \text{HOLD}$ initially holds $[\sigma_1]_i, \dots, [\sigma_B]_i$. Party P_i interprets these B values as evaluations at points $1, \dots, B$ of a polynomial of degree at most $(B - 1)$. It then evaluates this polynomial at the remaining points and thereby derives $[\sigma_j]_i$ for $j \in [B + 1, n]$. Each $P_i \in \text{HOLD}$ will send $[\sigma_j]_i$ to P_j for every $j \in [n]$.

For each $j \in [n]$, party P_j receives $n - t$ shares from **HOLD**, of which at most t may be incorrect. Since $n - 2t \geq d' + t + 1$, P_j can uniquely decode a degree- d' polynomial $\mathbf{Y}_j$. It can therefore recover the authentic signature $\sigma_j = (\mathbf{Y}_j(0), \dots, \mathbf{Y}_j(1 - a))$.

In the second stage, each P_j for $j \in [n]$ broadcasts its reconstructed σ_j . At least $n - t$ of these broadcasts originate from honest parties. For each $v \in [a]$, the n values $\mathbf{Y}_j(1 - v)$ for $j \in [n]$ form evaluations of a polynomial (denoted by $\mathbf{P}_v$) of degree at most $(B - 1)$. Therefore, if $n - t \geq (B - 1) + 1 + t$, the parties can uniquely decode $\mathbf{P}_v$ despite up to t erroneous broadcasts. Consequently, all honest parties recover the authentic signatures $\sigma_1, \dots, \sigma_B$, as desired. Importantly, this reconstruction requires only $\mathcal{O}(n)$ field operations per party, in contrast to the $\mathcal{O}(n)$ group operations performed by the aggregator in **SPRINT**. Applying the above online reconstruction protocol to **DASH2** under the stronger honest majority assumption yields our **DASH3** protocol.

2 Preliminaries

2.1 Basic Notation

We consider n parties $P_1, \dots, P_n$, of which at most t may be corrupted. For a positive integer n , we write $[n] = \{1, \dots, n\}$. For integers $a \leq b$, the interval $[a, b]$ denotes the set $\{a, \dots, b\}$. We use **hash** to denote a hash function modeled as a random oracle.

Let $\mathbb{F}_p$ be the finite field of prime order p . The notation $x \xleftarrow{\$} \mathbb{F}_p$ denotes uniform sampling from $\mathbb{F}_p$. Vectors are written in bold lowercase letters (e.g., $\mathbf{v}$).

and matrices in uppercase letters (e.g., X). For a matrix X , the entry in row i and column j is denoted by $X_{i,j}$. We write X_S for the submatrix formed by the rows or columns indexed by S , with the choice of rows or columns determined by context.

For a non-empty set of interpolation points $\mathcal{A} \subseteq \mathbb{F}_p$ and an evaluation point $i \in \mathbb{F}_p$, define the Lagrange coefficient $\lambda_{i,v} = \prod_{j \in \mathcal{A} \setminus \{v\}} (i - j)(v - j)^{-1}$. These coefficients satisfy $f(i) = \sum_{v \in \mathcal{A}} \lambda_{i,v} f(v)$ for every polynomial $f \in \mathbb{F}_p[X]$ of degree less than $|\mathcal{A}|$.

2.2 Network and Adversarial Model

We consider n parties communicating over an asynchronous network, where message delivery has no known time bound. A slow honest party is therefore indistinguishable from a corrupted party that withholds its messages. Protocols proceed after receiving sufficiently many messages without waiting for all n parties.

We assume access to authenticated point-to-point channels and a total-order broadcast channel. The latter is motivated by blockchain deployments in which the underlying consensus layer provides a globally ordered broadcast service. It satisfies the following properties:

1. **Complete delivery:** If an honest party broadcasts a message, or if any honest party delivers one, then all honest parties will eventually deliver the same message.
2. **Authenticity:** If an honest party delivers a message claimed to be from an honest party, then that message was indeed broadcast by the claimed sender.
3. **Prefix consistency:** At any time, the sequence of messages delivered by one honest party is a prefix of the sequence delivered by any other honest party, or vice versa.

Unless stated otherwise, broadcasting refers to the total-order broadcast channel. We assume that every message sent between honest parties is eventually delivered, while corrupted parties may deviate arbitrarily from the protocol. We consider a static malicious adversary, where the corrupted set is fixed before execution. Robustness ensures that adversarial behavior neither causes honest parties to accept an invalid output nor leads to a permanent abort.

2.3 Player Elimination

Player elimination [6,24] is a general way to turn an efficient but non-robust protocol into a robust one with only a small amortized overhead. It starts from a detectable protocol in which an execution may produce an incorrect result in the presence of malicious parties, but any such result is detected by at least one honest party. This technique works as follows:

- If the execution succeeds, the parties keep the output and terminate.

- If the detectable stage fails, the parties identify a pair guaranteed to contain at least one corrupted party, remove both parties from the active party set, and repeat the affected stage with updated parameters.

To generate signing nonces, each party acts as a dealer and distributes encrypted shares of its random contribution, along with public commitments that allow the receivers (shareholders) to check those shares. In our basic construction, a shareholder can report an invalid share with publicly verifiable evidence against the dealer.

Our optimized constructions check many shares together and omit this evidence to reduce the cost of handling complaints. A complaint therefore leaves open whether the dealer supplied inconsistent data or the shareholder made a false accusation. Since an honest shareholder never complains against an honest dealer, removing both parties eliminates at least one corrupted party. The remaining parties repeat the complaint procedure using the shares and commitments already distributed, with at most t rounds ending in elimination.

2.4 Various Algebraic Computations

We use the terminology in [23,32] to distinguish algebraic operations whose costs differ substantially in practice. Let $\lambda = \lceil \log_2 p \rceil$ be the bit length of a field element. We write $\Gamma = \{0, \dots, 2^\eta - 1\} \subseteq \mathbb{F}_p$ for the set from which the public random coefficients used in batched checks are sampled, where η is the challenge bit length.³

- **Lazy scalar addition.** This is the addition or subtraction of two representatives in $[-p, p)$, followed by a reduction modulo p back to $[-p, p)$. It is used in finite difference computations for polynomial evaluation.
- **Tiny scalar/scalar multiplication.** This is the computation of γx , where $\gamma \xleftarrow{\$} \Gamma$ is public and $x \in \mathbb{F}_p$ is a secret, full-sized scalar. When many such products occur in a long sum, the amortized cost per product is less than 1/40 of one group addition for typical parameters.
- **Full scalar/scalar multiplication.** This is the multiplication of two full-sized elements of $\mathbb{F}_p$. When the products occur only as terms in a long sum, reduction modulo p can be delayed until after the sum is accumulated.
- **Tiny scalar/group multiplication.** This is the computation of $\gamma \cdot X$, where $\gamma \xleftarrow{\$} \Gamma$ is public and $X \in \mathbb{G}$ is variable and public. When many such products occur in a long sum, the amortized cost per product is only one or two group additions for typical parameters.
- **Full scalar/group multiplication.** This is the computation of $x \cdot G$, where $x \in \mathbb{F}_p$ is a secret, full-sized scalar and G is the fixed group generator. Precomputation on G may be used, but the implementation must run in constant time because x is secret. For typical parameters, this operation requires approximately 43 group additions and six doublings.

³ See Appendix B.2 for the choice of η .

3 DASH1: The Base Construction

We present a robust threshold signature protocol that follows the standard offline–online paradigm. The expensive generation and verification of ephemeral randomness take place in the offline phase, while the online phase requires only the generation and aggregation of signature shares. Our construction in this section operates over an asynchronous network with up to t corrupted parties and assumes that $n \geq 3t + 2a - 1$, where a denotes the packing factor. To improve efficiency, the protocol processes ephemeral randomness in batches and distributes the associated computation among the parties.

Section 3.1 gives an overview of the main protocol, including the generation of shared nonces and signature shares. The agreement and randomness extraction subprotocols are described in Sections 3.2 and 3.3, respectively.

3.1 The Main Protocol

Setup phase. This phase generates a degree- d packed sharing $[\mathbf{s}]$ of $\mathbf{s} = (s, \dots, s) \in \mathbb{F}_p^a$, together with the public verification key $\mathbf{vk} = s \cdot G$. Each party P_i obtains a share $\mathbf{s}_i = \mathbf{F}(i)$. The corresponding verification key share $\mathbf{vk}_i = \mathbf{F}(i) \cdot G$ is publicly available.

The setup phase is executed only once and reused across many signing batches. Before nonce generation begins, every honest party must hold a valid long-term key share. This is necessary because the nonce generation phase selects a changing set **HOLD** based only on nonce share validity, and each $P_i \in \mathbf{HOLD}$ later combines its nonce share with its long-term key share $[\mathbf{s}]_i$ to produce a signature share. If an honest party in **HOLD** lacks a valid long-term share, the number of valid signature shares may fall below the reconstruction threshold. We therefore instantiate the setup phase with a complete asynchronous DKG protocol, such as [23]. Its completeness guarantees that every honest party eventually obtains a valid key share and that all honest parties agree on the public key and the corresponding public key shares.

Robust batched nonce generation. We say that a threshold signature protocol is robust, or provides guaranteed output delivery, if every signing request eventually produces a valid signature despite arbitrary deviations by corrupted parties. In the offline phase of our construction, the parties agree on common sets of dealers and shareholders and produce sufficiently many usable shared nonces, while ensuring that every honest shareholder selected for signing holds valid shares from every qualified dealer. In the online phase, the parties reconstruct the corresponding signatures without waiting for all parties to respond.

To generate ephemeral randomness, each party P_i acts as a dealer and chooses L random degree- d' polynomials $\{\mathbf{H}_i^{(\ell)}\}_{\ell \in [L]}$. For each polynomial, the dealer publishes commitments to its evaluations at $d' + 1$ public points. For every shareholder P_j , it computes the share $s_{i,j}^{(\ell)} = \mathbf{H}_i^{(\ell)}(j)$, encrypts it as $\text{ct}_{i,j}^{(\ell)}$, and broadcasts the ciphertext. Each P_j decrypts its ciphertext and checks the resulting

share against the public commitments. If a check fails, P_j can reveal the disputed share together with a proof of correct decryption, allowing everyone to verify the complaint.

The agreement protocol turns these local checks into a common decision. It outputs three common sets: **QUAL** contains the dealers whose contributions are retained, **BAD** contains the excluded dealers, and **HOLD** contains the shareholders selected for later signing. The output satisfies $|\mathbf{HOLD}| = n - t$ and $|\mathbf{QUAL}| + |\mathbf{BAD}| = n - t$, and every honest shareholder in **HOLD** has valid shares from all dealers in **QUAL**. Section 3.2 explains how the parties obtain these common sets.

Agreement ensures that the retained contributions are consistently shared, but contributions from corrupted dealers need not be uniformly random. We therefore use randomness extraction to obtain independent, uniformly random shared nonces from the honest dealers' randomness without identifying which dealers are honest. Since every dealer in **BAD** has been proven to be corrupt, **QUAL** contains at least $b = |\mathbf{QUAL}| - (t - |\mathbf{BAD}|) = n - 2t$ honest dealers. The parties apply a $b \times |\mathbf{QUAL}|$ super-invertible matrix to the corresponding shares and commitments to derive b packed nonce sharings per batch label and their partial signatures.

Online signing with packed shares. Once the offline phase has produced the shared nonces, signing requires only local computation and one broadcast from each shareholder involved. At the start of the online phase, the parties hash the public information to obtain a common value δ and set $\Delta = \delta G$. Following the method of [22] and its batched version in [9], they use this common value to randomize the presignatures so that multiple messages can be signed securely in parallel. The parties then compute the challenge for each message using the new partial signature.

Packed secret sharing lets each shareholder contribute to all a signatures with a single signature share. Omitting the batch indices, let $\mathbf{F}$ and $\mathbf{H}$ be the polynomials sharing the key and nonces of degrees $d = t + a - 1$ and $d' = t + 2a - 2$, respectively. Encoding the public challenges in a degree- $(a - 1)$ polynomial $\mathbf{Z}$ gives the signature polynomial $\mathbf{Y} = \mathbf{H} + \mathbf{Z}\mathbf{F}$, whose degree is at most d' . Each $P_i \in \mathbf{HOLD}$ locally computes $\mathbf{Y}(i)$ from its nonce share $\mathbf{H}(i)$, key share $\mathbf{F}(i)$ and public value $\mathbf{Z}(i)$, and broadcasts it.

Anyone can verify these shares using the public nonce commitments and the shareholders' public key shares, the invalid shares are then discarded. Since **HOLD** contains $n - t$ shareholders, at most t of whom are corrupted, at least $n - 2t$ valid shares eventually arrive. The $n \geq 3t + 2a - 1$ setting ensures that this number is at least $d' + 1$, so an aggregator can reconstruct $\mathbf{Y}$ from valid shares of distinct shareholders. Evaluating $\mathbf{Y}$ at the a packing points recovers the responses $r + e \cdot s$ for all a messages. Adding δ to each response and pairing it with the corresponding partial signature gives the Schnorr signature. The share verification and signature aggregation procedures are specified in Appendix A.1.

Protocol 1: Π_{DASH1}

Parameters: Integers $n, t, a \geq 1$ and a batch size $L \geq n$, with $n \geq 3t + 2a - 1, d = t + a - 1, d' = t + 2a - 2$ and a finite field $\mathbb{F}_p$. Denote the party set as $\mathcal{P} = \{P_1, \dots, P_n\}$.

Setup.

1. Let $\mathbf{s} = (s, \dots, s) \in \mathbb{F}_p^a$. The parties hold a degree- d packed sharing $[\mathbf{s}]$ defined by a random polynomial $\mathbf{F}$ satisfying $\mathbf{F}(1 - v) = \mathbf{s}$ for $v \in [a]$, where each P_i holds $[\mathbf{s}]_i = \mathbf{F}(i)$.
2. Define the public keys as $\text{vk} = \mathbf{s} \cdot G$ and $\text{vk}_i = \mathbf{F}(i) \cdot G$.

Ephemeral Randomness Generation. Each party P_i performs the following steps.

1. Party P_i chooses random degree- d' polynomials $\{\mathbf{H}_i^{(\ell)}\}_{\ell \in [L]}$. For every $v \in [1 - a, d' - a + 1], \ell \in [L]$ and $j \in [n]$, P_i computes $\hat{\mathbf{H}}_i^{(\ell)}(v) = \mathbf{H}_i^{(\ell)}(v) \cdot G$ and encrypts $s_{i,j}^{(\ell)} = \mathbf{H}_i^{(\ell)}(j)$ under P_j 's public encryption key as $\text{ct}_{i,j}^{(\ell)}$. It broadcasts the commitments and ciphertexts.
2. The parties invoke $\Pi_{\text{Agreement}}$ on these commitments and ciphertexts, obtain common sets (HOLD, QUAL, BAD) such that $|\text{HOLD}| = n - t, |\text{QUAL}| + |\text{BAD}| = n - t$, and every honest $P_i \in \text{HOLD}$ holds valid shares from all dealers in QUAL.
3. Let $b = n - 2t^1$ and $c = |\text{QUAL}|$. Let $\Phi \in \mathbb{F}_p^{b \times c}$ be the super-invertible matrix. Order the dealers as $\text{QUAL} = (P_{Q_1}, \dots, P_{Q_c})$ and proceed as follows.
 - For every $v \in [a], \ell \in [L]$ and $j \in [c]$, define the public matrix X by $X_{j,(v,\ell)} = \hat{\mathbf{H}}_{Q_j}^{(\ell)}(1 - v)$. The parties invoke Π_{SIM} with $(N, f) = (n, t)$ and $\mathcal{V} = [a]$ to obtain retained labels $T \subseteq [L]$ and the partial signatures $R^{(u,v,\ell)} = (\Phi X)_{u,(v,\ell)}$ for each $u \in [b], v \in [a]$ and $\ell \in T$.
 - For every $u \in [b]$ and $\ell \in [L]$, define $\mathbf{H}^{(u,\ell)} = \sum_{j \in [c]} \Phi_{u,j} \mathbf{H}_{Q_j}^{(\ell)}$ and $\mathbf{r}^{(u,\ell)} = (\mathbf{H}^{(u,\ell)}(1 - v))_{v \in [a]}$. For each $u \in [b]$ and $\ell \in T$, every $P_i \in \text{HOLD}$ computes $[\mathbf{r}^{(u,\ell)}]_i = \sum_{j \in [c]} \Phi_{u,j} s_{Q_j,i}^{(\ell)}$.

Signature Share Generation. On input messages $M^{(u,v,\ell)}, u \in [b], v \in [a], \ell \in T$: Let $\delta = \text{hash}(\text{vk}, \text{QUAL}, \{M^{(u,v,\ell)}, R^{(u,v,\ell)} : u \in [b], v \in [a], \ell \in T\})$ and $\Delta = \delta \cdot G$. Each $P_i \in \text{HOLD}$ performs the following steps for every $u \in [b], \ell \in T$.

1. Compute the challenges $e^{(u,v,\ell)} = \text{hash}(\text{vk}, \Delta + R^{(u,v,\ell)}, M^{(u,v,\ell)})$ for all $v \in [a]$.
2. Compute the degree- $(a - 1)$ packed encoding polynomial $\mathbf{Z}^{(u,\ell)}$ satisfying $\mathbf{Z}^{(u,\ell)}(1 - v) = e^{(u,v,\ell)}$ for $v \in [a]$.

3. Compute the packed signature share $[\sigma^{(u,\ell)}]_i = [\mathbf{r}^{(u,\ell)}]_i + \mathbf{Z}^{(u,\ell)}(i) \cdot [\mathbf{s}]_i^2$.

Signature Aggregation. All parties invoke Π_{Online} in Appendix A.1 to output the corresponding Schnorr signatures.

¹ Since $|\text{QUAL}| + |\text{BAD}| = n - t$, at least $n - 2t$ of whom are honest. Every dealer in **BAD** is corrupted, so all these honest dealers belong to **QUAL**.

² $[\sigma^{(u,\ell)}]_i = \mathbf{Y}^{(u,\ell)}(i)$ for the degree- d' polynomial $\mathbf{Y}^{(u,\ell)} = \mathbf{H}^{(u,\ell)} + \mathbf{Z}^{(u,\ell)} \cdot \mathbf{F}$.

3.2 The Agreement Protocol

Although each shareholder can check its shares against a dealer's commitments, a corrupted dealer may send valid shares to some honest shareholders and invalid shares to others, causing disagreement over which contributions to retain. The parties must therefore agree on the contributing dealers while ensuring that the honest shareholders selected for signing hold valid shares from all of them. The agreement protocol $\Pi_{\text{Agreement}}$ meets these requirements by jointly selecting common sets of dealers and signing shareholders.

Using the common broadcast order, the parties select the first $n - t$ distinct senders of well-formed dealer messages as candidate dealers. A *well-formed dealer message* has the prescribed format and contains all commitments and ciphertexts required for the current batch, but well-formedness alone does not guarantee that the underlying shares match the commitments. Each shareholder therefore decrypts its ciphertexts from the candidate dealers and checks the resulting shares against the corresponding commitments before broadcasting a single complaint message, which is empty if all checks pass. Every complaint includes the disputed share and a proof of correct decryption, allowing all parties to verify the alleged inconsistency.

The first $n - t$ distinct senders of valid complaint messages form **HOLD**, the set of shareholders selected for signing. Any candidate dealer named in a verifiable complaint in one of these messages is placed in **BAD**, while the remaining candidates form **QUAL** and contribute to nonce generation. Prefix consistency of the total-order broadcast channel, ensures that all honest parties select the same candidate dealers and complaint messages, while public verification of complaints gives them the same sets **BAD** and **QUAL**. Moreover, a dealer cannot remain in **QUAL** if an honest shareholder in **HOLD** has complained against it, so every such shareholder holds valid shares from all retained dealers.

Protocol 2: $\Pi_{\text{Agreement}}$

Precondition: The parameters n, t, a, L are integers with $n \geq 3t + 2a - 1$, and the nonce polynomial degree is $d' = t + 2a - 2$. The protocol takes as input the commitments and encrypted shares broadcast during ephemeral

randomness generation. The parties communicate over the total-order broadcast channel.

Initialize $\text{HOLD}, \text{QUAL}, \text{BAD} \leftarrow \emptyset$. Each party P_i proceeds as follows.

1. Let QUAL contain the first $n - t$ distinct senders of well-formed dealer messages.
2. Each P_i checks the shares received from every dealer $P_j \in \text{QUAL}$. For each $\ell \in [L]$, party P_i decrypts $\text{ct}_{j,i}^{(\ell)}$ to obtain $s_{j,i}^{(\ell)}$ and checks whether $s_{j,i}^{(\ell)} G = \sum_{v=1-a}^{d'-a+1} \lambda_{i,v} \hat{\mathbf{H}}_j^{(\ell)}(v)$. If any check for P_j fails, P_i forms a complaint containing the smallest failing index ℓ , the corresponding decrypted share, and a proof of correct decryption. Party P_i broadcasts all its complaints in one message, or the empty set if all checks pass.
3. Let HOLD contain the first $n - t$ distinct senders of valid complaint messages. For each $P_j \in \text{QUAL}$, add P_j to BAD if any of these messages contains a verifiable complaint against P_j . Then update $\text{QUAL} \leftarrow \text{QUAL} \setminus \text{BAD}$. Output $(\text{HOLD}, \text{QUAL}, \text{BAD})$.

Theorem 1. *Consider an execution of the base agreement protocol $\Pi_{\text{Agreement}}$ by n parties over a total-order broadcast channel. At most t parties are corrupted, and a is the packing parameter. Assume that no verifiable complaint is raised against an honest dealer. Every honest shareholder eventually terminates and outputs the same sets HOLD , QUAL , and BAD . These sets satisfy $|\text{HOLD}| = n - t$ and $|\text{QUAL}| + |\text{BAD}| = n - t$ and have the following properties.*

- No shareholder in HOLD complained against any dealer in QUAL .
- Every dealer in BAD is corrupted and was named in a verifiable complaint from a shareholder in HOLD .

In particular, every honest shareholder in HOLD holds valid shares from every dealer in QUAL .

The proof of Theorem 1 appears in Appendix B.1.

3.3 Distributed Randomness Extraction

The agreement protocol gives all honest parties the same sets QUAL , BAD , and HOLD . It determines which dealer contributions can be used and which shareholders will later provide signature shares. However, QUAL may still contain corrupted dealers, and the parties do not know which dealer contributions are honest. The remaining task is to combine all contributions in QUAL into fresh shared nonces despite this uncertainty.

To cover all three DASH constructions, we use N and f for the number of computation parties and their corruption bound in a given invocation. In DASH1, we have $(N, f) = (n, t)$ whereas in DASH2 and DASH3, $(N, f) = (n', t')$ refers to the remaining parties after player elimination. The agreement guarantees imply that QUAL contains at least $b = N - 2f$ honest dealers. Let $c = |\text{QUAL}|$ and order the qualified dealers as $\text{QUAL} = (P_{Q_1}, \dots, P_{Q_c})$.

Extracting the partial signatures. For each batch label ℓ , every dealer in QUAL contributes one packed nonce polynomial with a signing slots. We arrange the commitments to these values in a matrix $X \in \mathbb{G}^{c \times aL}$, with rows indexed by qualified dealers and columns by pairs $(v, \ell) \in [a] \times [L]$. Specifically, $X_{j,(v,\ell)} = \hat{\mathbf{H}}_{Q_j}^{(\ell)}(1-v)$ is dealer P_{Q_j} 's commitment to the nonce at packing point $1-v$ for label ℓ . The a columns for each label are consecutive and form a block.

At least b of the c qualified contributions come from honest dealers, each sharing aL independent uniform nonces. These contributions provide enough randomness to extract at least b packed nonce sharings per label. We multiply X on the left by a $b \times c$ super-invertible matrix Φ^4 , any b columns of which form an invertible matrix. The resulting partial signatures are given by ΦX^5 , where

$$(\Phi X)_{u,(v,\ell)} = \sum_{j=1}^c \Phi_{u,j} \hat{\mathbf{H}}_{Q_j}^{(\ell)}(1-v) = \hat{\mathbf{H}}^{(u,\ell)}(1-v).$$

Thus, row u of the block for label ℓ commits to the a nonce values packed in $\mathbf{H}^{(u,\ell)} = \sum_{j=1}^c \Phi_{u,j} \mathbf{H}_{Q_j}^{(\ell)}$. Each shareholder $P_i \in \text{HOLD}$ applies the same coefficients to its private shares, obtaining $\mathbf{H}^{(u,\ell)}(i) = \sum_{j=1}^c \Phi_{u,j} s_{Q_j,i}^{(\ell)}$, a share of the packed nonces committed to in that row.

Distributing and verifying the computation. Both Φ and X are public, so each party can compute ΦX locally. Having every party evaluate the full product, however, would duplicate the extraction work. Protocol Π_{SIM} therefore partitions the batch labels $[L]$ as evenly as possible into disjoint sets $\mathcal{L}_1, \dots, \mathcal{L}_N$. Writing X_i for the columns of X indexed by $[a] \times \mathcal{L}_i$, party P_i computes and broadcasts its assigned block $\tilde{R}_i = \Phi X_i$.

After receiving a claimed block $\tilde{R}_j$, party P_i privately samples a uniformly random row vector $\lambda^{(i,j)} \in \mathbb{F}_p^b$ and checks whether $\lambda^{(i,j)} \tilde{R}_j = (\lambda^{(i,j)} \Phi) X_j$. The left side combines the b rows of the received block into one row using the random coefficients in $\lambda^{(i,j)}$. To compute the right side, the verifier first combines the rows of Φ using the same coefficients and then multiplies the resulting single row by X_j . A correct block satisfies the equality by associativity of matrix multiplication, whereas an incorrect block passes with probability at most $1/p$. Moreover, the random vector remains private and the check requires no additional communication.

Each party processes blocks in the common broadcast order until it has accepted blocks from $N - f$ distinct computation parties. It outputs the union T of their assigned label sets together with the corresponding partial signatures. Provided that no incorrect block is accepted, all honest parties obtain the same

⁴ A super-invertible matrix can be constructed as $\Phi = (I_b \mid \Psi)$ from any $b \times (c-b)$ hyper-invertible matrix Ψ , whose square submatrices are all invertible. The identity block requires no scalar/group multiplications, reducing the direct evaluation cost per column from bc to $b(c-b)$ such multiplications.

⁵ The ECFFT EXTEND algorithm of [8] evaluates this product using $O(aLk \log k)$ full scalar/group multiplications, where k is a power of two satisfying $k \geq \max(b, c-b)$.

label set T and partial signatures. At least $N - f$ honest parties eventually broadcast correct blocks, so corrupted parties cannot prevent termination by withholding their outputs.

Protocol 3: Π_{SIM}

Precondition: The protocol is run by N parties, of which at most f are corrupted. The public data consist of a batched label set $\mathcal{L}$ with $|\mathcal{L}| \geq N$, an index set $\mathcal{V}$ for the packing points, a matrix $\Phi \in \mathbb{F}_p^{b \times c}$, and a matrix $X \in \mathbb{G}^{c \times (|\mathcal{V}| |\mathcal{L}|)}$ whose rows follow the agreed order of the qualified dealers and whose columns are indexed by $\mathcal{V} \times \mathcal{L}$.

1. Publicly partition $\mathcal{L}$ as evenly as possible into disjoint sets $\{\mathcal{L}_i\}_{i \in [N]}$.
2. Each party P_i computes its assigned block $\tilde{R}_i = \Phi X_{[\cdot, \mathcal{V} \times \mathcal{L}_i]} \in \mathbb{G}^{b \times (|\mathcal{V}| |\mathcal{L}_i|)}$ and broadcasts it.
3. Upon receiving the first claimed block $\tilde{R}_j$ from P_j , party P_i samples $\lambda^{(i,j)} \xleftarrow{\$} \mathbb{F}_p^b$ and accepts the block if $\lambda^{(i,j)} \tilde{R}_j = (\lambda^{(i,j)} \Phi) X_{[\cdot, \mathcal{V} \times \mathcal{L}_j]}$. Otherwise, it rejects the block and ignores later blocks from P_j .
4. Upon accepting the first $N - f$ valid blocks, let T be the union of the label sets assigned to their senders. Output T and the accepted entries of ΦX whose columns are indexed by $\mathcal{V} \times T$.

Theorem 2. *Consider an execution of Π_{SIM} by N computation parties over the total-order broadcast channel. At most f parties are statically corrupted, where $0 \leq f < N/2$. Except with probability at most $f \cdot (N - f)/p$, the following statements hold simultaneously.*

- For a super-invertible matrix Φ , any column of ΦX is uniform over $\mathbb{G}^b$ if the corresponding column of X contains b jointly uniform entries independent of the remaining entries.
- All honest parties select the same $N - f$ accepted blocks and output the same set $T \subseteq \mathcal{L}$ and the same matrix of group elements whose rows are indexed by $[b]$ and whose columns are indexed by $\mathcal{V} \times T$.
- For every $u \in [b]$ and $(v, \ell) \in \mathcal{V} \times T$, the common output equals $(\Phi X)_{u, (v, \ell)}$.

The proof of Theorem 2 appears in Appendix C.1.

4 DASH2: Batched Verification with Player Elimination

In the base construction, each shareholder verifies its share of every nonce polynomial separately, even though presignatures are generated in batches. Each check requires group multiplications to combine the polynomial's $d' + 1$ commitments. To reduce this cost, we batch the consistency checks across nonce polynomials, and use player elimination to address the resulting complications in complaint handling. The resulting protocol retains the condition $n \geq 3t + 2a - 1$ and the offline-online structure of the base construction.

4.1 The Main Protocol

The cost of individual verification. In the base construction Π_{DASH1} , each dealer broadcasts $d' + 1$ commitments per nonce polynomial. To verify a share from dealer P_i , shareholder P_j first interpolates these commitments to compute $\hat{\mathbf{H}}_i^{(\ell)}(j) = \sum_{v=1-a}^{d'-a+1} \lambda_{j,v} \hat{\mathbf{H}}_i^{(\ell)}(v)$, the group element corresponding to the expected share. It accepts the decrypted share $s_{i,j}^{(\ell)}$ only if $s_{i,j}^{(\ell)} \cdot G = \hat{\mathbf{H}}_i^{(\ell)}(j)$. Repeating this check for L shares from each of c candidate dealers requires $\mathcal{O}(cd'L)$ full scalar/group multiplications per shareholder, in addition to decryption. This cost motivates checking the L sharings from each dealer together, rather than verifying each one separately.

Batched consistency checks. We use the batched consistency checks of [23] to verify these polynomials together. Besides its L nonce polynomials, each dealer P_i chooses n independent degree- d' blinding polynomials $\{\mathbf{G}_i^{(m)}\}_{m \in [n]}$, broadcasts commitments and encrypted shares for all these polynomials. After this message has been fixed, a random beacon protocol Π_{Beacon}^i supplies independent coefficients $\{\gamma_i^{(\ell,m)}\}_{\ell \in [L], m \in [n]} \in \Gamma^{L \times n}$, where Γ is a subset of $\mathbb{F}_p$. The beacon can be instantiated using AVSS and consensus to select $t + 1$ sharings, reconstruct their secrets and sum them. Alternatively, it can be based on threshold BLS signatures [11,10]. A PRG or hash function expands the beacon output into the required coefficients. For every $m \in [n]$, the dealer then opens the response polynomial

$$\mathbf{C}_i^{(m)} = \mathbf{G}_i^{(m)} + \sum_{\ell \in [L]} \gamma_i^{(\ell,m)} \mathbf{H}_i^{(\ell)}.$$

Here, $\mathbf{G}_i^{(m)}$ masks the linear combination of nonce polynomials, allowing the dealer to open $\mathbf{C}_i^{(m)}$ without revealing the nonces.

Each shareholder P_j checks that every response polynomial has degree at most d' and compares its evaluation at point j with the same linear combination of the received nonce and blinding shares. Specifically, for every $m \in [n]$, it checks whether $\mathbf{C}_i^{(m)}(j) = w_{i,j}^{(m)} + \sum_{\ell \in [L]} \gamma_i^{(\ell,m)} s_{i,j}^{(\ell)}$ holds, where $s_{i,j}^{(\ell)}$ and $w_{i,j}^{(m)}$ denote its decrypted nonce and blinding shares, respectively. We call these *private checks*, as they use the shareholder's decrypted shares and require only field operations.

It also performs *public checks*, which compare the response indexed by j with the public commitments. For every point $v \in \mathcal{V} = [1 - a, d' - a + 1]$, it verifies $\mathbf{C}_i^{(j)}(v) \cdot G = \hat{\mathbf{G}}_i^{(j)}(v) + \sum_{\ell \in [L]} \gamma_i^{(\ell,j)} \hat{\mathbf{H}}_i^{(\ell)}(v)$. These public checks require $|\mathcal{V}|L$ tiny scalar/group multiplications and $|\mathcal{V}|$ full scalar/group multiplications. Together, the private and public checks test the consistency of the received nonce shares with the published commitments. The challenge set Γ must be chosen so that the cumulative error is negligible.

Handling failed checks. Consider the case where shareholder P_j finds that dealer P_i 's response $\mathbf{C}_i^{(m)}(j)$ does not match the linear combination of P_j 's decrypted nonce and blinding shares. This private check combines all L nonce

shares, so a mismatch does not identify a particular invalid share and may instead be caused by an incorrect blinding share or response. Moreover, other parties cannot verify the mismatch from the public transcript alone, since it depends on P_j 's decrypted shares. A publicly verifiable complaint would therefore require evidence about these values and may require additional checks to locate an invalid share. Our agreement protocol allows P_j to name P_i in its complaint without attaching this evidence. Other parties then cannot tell whether P_i supplied inconsistent data or P_j made a false accusation. The protocol uses player elimination to resolve this dispute by removing both parties from the active party set. Since an honest shareholder never complains against an honest dealer, every removed pair contains at least one corrupted party.

After removing disputed pairs, the parties must still have enough valid shares for randomness extraction and signing. We therefore reuse the dealer messages and repeat only the complaint rounds, checking additional dealers as needed. Section 4.2 shows how removing disjoint pairs preserves the required number of honest qualified dealers and valid signature shares.

Protocol 4: Π_{DASH2}

Parameters: Integers $n, t, a \geq 1$ satisfying $n \geq 3t + 2a - 1$, and a batch size $L \geq n$, with degrees $d = t + a - 1$ and $d' = t + 2a - 2$. Let $\mathbb{F}_p$ be a finite field and let $\Gamma \subseteq \mathbb{F}_p$. Denote the party set by $\mathcal{P} = \{P_1, \dots, P_n\}$.

Setup. The step here is the same as in Protocol Π_{DASH1} .

Ephemeral Randomness Generation. Each party $P_i \in \mathcal{P}$ performs the following steps.

1. Choose L random degree- d' nonce polynomials $\{\mathbf{H}_i^{(\ell)}\}_{\ell \in [L]}$ and n random degree- d' blinding polynomials $\{\mathbf{G}_i^{(m)}\}_{m \in [n]}$. For every $v \in [1 - a, d' - a + 1]$, $\ell \in [L]$ and $m \in [n]$, compute $\hat{\mathbf{H}}_i^{(\ell)}(v) = \mathbf{H}_i^{(\ell)}(v) \cdot G$ and $\hat{\mathbf{G}}_i^{(m)}(v) = \mathbf{G}_i^{(m)}(v) \cdot G$. For every $P_j \in \mathcal{P}$, encrypt under P_j 's public encryption key the shares $\{s_{i,j}^{(\ell)} = \mathbf{H}_i^{(\ell)}(j)\}_{\ell \in [L]}$ and $\{w_{i,j}^{(m)} = \mathbf{G}_i^{(m)}(j)\}_{m \in [n]}$, and broadcast all commitments and ciphertexts. Upon receiving a broadcast message from $P_j \in \mathcal{P}$, P_i initializes Π_{Beacon}^j .
2. Wait for Π_{Beacon}^i to deliver $\{\gamma_i^{(\ell,m)}\}_{\ell \in [L], m \in [n]} \in \Gamma^{L \times n}$. For every $m \in [n]$, compute and broadcast $\mathbf{C}_i^{(m)} = \hat{\mathbf{G}}_i^{(m)} + \sum_{\ell \in [L]} \gamma_i^{(\ell,m)} \cdot \hat{\mathbf{H}}_i^{(\ell)}$.
3. Set $\mathcal{V} = [1 - a, d' - a + 1]$. The parties invoke $\Pi'_{\text{Agreement}}$ on the data generated above to obtain $(\mathcal{P}', n', t', \text{HOLD}, \text{QUAL})$, where $\mathcal{P}'$ is the remaining party set.
4. Let $b = n' - 2t' = n - 2t^1$ and $c = |\text{QUAL}| = n' - t'$. Let $\Phi \in \mathbb{F}_p^{b \times c}$ be the super-invertible matrix. Order the dealers as $\text{QUAL} = (P_{Q_1}, \dots, P_{Q_c})$ and proceed as follows:
 - For every $v \in [a], \ell \in [L], j \in [c]$, define the public matrix X by $X_{j,(v,\ell)} = \hat{\mathbf{H}}_{Q_j}^{(\ell)}(1 - v)$. The parties in $\mathcal{P}'$ invoke Π_{SIM} with

- $(N, f) = (n', t')$ and $\mathcal{V} = [a]$, obtain retained labels $T \subseteq [L]$ and the partial signatures $R^{(u,v,\ell)} = (\Phi X)_{u,(v,\ell)}$ for each $u \in [b]$, $v \in [a]$ and $\ell \in T$.
- For every $u \in [b], \ell \in [L]$, define $\mathbf{H}^{(u,\ell)} = \sum_{j \in [c]} \Phi_{u,j} \mathbf{H}_{Q_j}^{(\ell)}$, $\mathbf{r}^{(u,\ell)} = (\mathbf{H}^{(u,\ell)}(1-v))_{v \in [a]}$. For each $u \in [b]$ and $\ell \in T$, every $P_i \in \text{HOLD}$ computes $[\mathbf{r}^{(u,\ell)}]_i = \sum_{j \in [c]} \Phi_{u,j} s_{Q_j,i}^{(\ell)}$.

Signature Share Generation. The step here is the same as in Protocol Π_{DASH1} .

Signature Aggregation. The step here is the same as in Protocol Π_{DASH1} .

¹ At most t' parties in **QUAL** are corrupted, so **QUAL** contains at least $|\text{QUAL}| - t' = n' - 2t' = n - 2t$ honest dealers.

A variation for large n . For large n , we can reduce the number of blinding and response polynomials following [23]. Choose $K \in [n]$, each dealer generates only K blinding polynomials and opens K response polynomials, using independent beacon challenges for each response. Each shareholder P_j checks all K responses against its private shares. For the public checks, shareholder P_j uses the response indexed by $m_j = (j-1) \bmod K + 1$. For each dealer P_i , it verifies

$$\mathbf{C}_i^{(m_j)}(v) \cdot G = \hat{\mathbf{G}}_i^{(m_j)}(v) + \sum_{\ell \in [L]} \gamma_i^{(\ell, m_j)} \hat{\mathbf{H}}_i^{(\ell)}(v), \quad v \in \mathcal{V}.$$

Thus, several shareholders may use the same response for their public checks. Using K blinding polynomials instead of n reduces each dealer's blinding commitments from $n(d'+1)$ to $K(d'+1)$ group elements and its encrypted blinding shares from n^2 to nK ciphertexts. The response payload also decreases from $n(d'+1)$ to $K(d'+1)$ field elements. Each shareholder performs K private checks per dealer instead of n , while still performing $|\mathcal{V}|$ public checks. Since shareholders assigned to the same response use the same challenge coefficients, a smaller K may require a larger challenge set Γ to keep the soundness error negligible.

4.2 The Agreement Protocol

The batched checks give each shareholder a local decision about each dealer. The agreement protocol turns these decisions into common sets **QUAL** and **HOLD**, where every honest shareholder in **HOLD** holds valid shares from every dealer in **QUAL**, except with the soundness error of the checks. Unlike the base agreement protocol $\Pi_{\text{Agreement}}$, it does not require the parties to identify which member of a disputed pair is corrupted.

Player elimination. The agreement protocol in this section proceeds in complaint rounds and resolves disputes by removing pairs of parties. Let $\mathcal{P} = \{P_1, \dots, P_n\}$ denote the original party set. A party is *active* if it has not yet been removed, and $\mathcal{P}'$ denotes the set of all such parties. Only active parties are eligible for selection as qualified dealers or as shareholders in **HOLD**. We write $n' = |\mathcal{P}'|$ for the number of active parties and t' for their corruption bound. Initially, all parties are active, so $\mathcal{P}' = \mathcal{P}$, $n' = n$, and $t' = t$.

Using the common broadcast order, the parties select the first $n' - t'$ well-formed dealer response messages from distinct active dealers and place their senders in **QUAL**. Each active shareholder performs the private and public checks on the response polynomials of every dealer in **QUAL**. It then broadcasts a complaint list, which is empty if all checks pass. The first $n' - t'$ valid complaint messages from distinct active shareholders for the current round determine **HOLD**. If all selected complaints are empty, the protocol terminates with these two sets.

Otherwise, the parties process the selected complaints in the common broadcast order. A valid complaint may not name its own sender. For each non-empty complaint, they proceed as follows.

- If the sender is no longer active, ignore the complaint.
- Otherwise, consider only the named dealers that are still active. If none remain, ignore the complaint.
- If at least one named dealer is still active, choose the one with the smallest index and remove both the dealer and the complaint sender from $\mathcal{P}'$.

Every removed pair contains at least one corrupted party, regardless of whether the complaint is true or false. After removing the pairs, the parties update n' and t' , remove inactive dealers from **QUAL**, and add further active dealers in the original order of response messages until $|\mathbf{QUAL}| = n' - t'$. Each remaining shareholder keeps its previous check results and checks only dealers newly added to **QUAL**, and then broadcasts a fresh complaint list.

An eliminated party sends no further complaint messages. If honest, it continues to follow the public transcript, initialize and run beacon instances for newly received dealer messages, so that the remaining dealers can obtain their challenges and produce response polynomials.

Validity and termination. If x pairs have been removed, then $n' = |\mathcal{P}'| = n - 2x$ and $t' = t - x$ is an upper bound on the number of remaining corrupted parties. This gives the identity $n' - 2t' = n - 2t$. Therefore, the qualified set of size $n' - t'$ still contains at least $n - 2t$ honest dealers, and a shareholder set of the same size still contains at least $n - 2t$ honest shareholders. The packing factor and polynomial degrees remain fixed throughout the agreement invocation, so this invariant preserves the number of valid shares needed for the original degree- d' polynomials.

Every non-terminating round removes at least one corrupted party, so there are at most t such rounds. At least $n' - t'$ honest parties remain active after each round. Their response and complaint messages eventually arrive, allowing the protocol to continue without waiting for corrupted parties. On termination, both

output sets have size $n' - t'$, and every honest shareholder selected in HOLD has approved every dealer in QUAL.

Protocol 5: $\Pi'_{\text{Agreement}}$

Precondition: A party set $\mathcal{P} = \{P_1, \dots, P_n\}$ with corruption bound t , together with the commitments, encrypted shares, response polynomials, and beacon challenges generated during ephemeral randomness generation. Here L is the batch size, d' is the degree of each nonce polynomial, and $\mathcal{V}$ is the set used for the public checks.

Initialize $\mathcal{P}' \leftarrow \mathcal{P}$, $n' \leftarrow n$, $t' \leftarrow t$, and $\text{HOLD}, \text{QUAL}, \text{BAD} \leftarrow \emptyset$. Each party proceeds as follows.

1. Set $\text{HOLD} \leftarrow \emptyset$ and $\text{QUAL} \leftarrow \text{QUAL} \cap \mathcal{P}'$. If $|\text{QUAL}| < n' - t'$, consider the well-formed response messages from additional active dealers in broadcast order and add their senders to QUAL until $|\text{QUAL}| = n' - t'$.
2. Each $P_i \in \mathcal{P}'$ checks only the response messages from dealers newly added to QUAL. For every such dealer P_j , P_i decrypts the shares it received from P_j and performs the following checks.
 - For every $P_m \in \mathcal{P}'$, check whether $\deg(\mathbf{C}_j^{(m)}) \leq d'$ and $\mathbf{C}_j^{(m)}(i) = w_{j,i}^{(m)} + \sum_{\ell \in [L]} \gamma_j^{(\ell,m)} \cdot s_{j,i}^{(\ell)}$ hold.
 - For every $v \in \mathcal{V}$, check whether $\mathbf{C}_j^{(i)}(v) \cdot G = \hat{\mathbf{G}}_j^{(i)}(v) + \sum_{\ell \in [L]} \gamma_j^{(\ell,i)} \cdot \hat{\mathbf{H}}_j^{(\ell)}(v)$ holds.

Each P_i broadcasts a complaint message for the current round, listing the dealers whose response messages fail at least one check. The dealers are listed in increasing order of party index. If no checks fail, P_i broadcasts an empty complaint message. A complaint message from P_i is valid only if every named party belongs to $\text{QUAL} \setminus \{P_i\}$.

3. Let HOLD contain the senders of the first $n' - t'$ valid complaint messages from the parties in $\mathcal{P}'$ that are bound to the current round. If all these messages are empty, output $(\mathcal{P}', n', t', \text{HOLD}, \text{QUAL})$ and terminate. Otherwise, process the selected non-empty complaints in broadcast order as follows.
 - If the complaint sender P_j no longer belongs to $\mathcal{P}'$, ignore the complaint. Otherwise, let A_j be the set of parties named in the complaint that still belong to $\mathcal{P}' \cap \text{QUAL}$. If $A_j = \emptyset$, ignore the complaint.
 - Otherwise, let P_h be the party with the smallest index in A_j , and set $\mathcal{P}' \leftarrow \mathcal{P}' \setminus \{P_j, P_h\}$ and $\text{BAD} \leftarrow \text{BAD} \cup \{P_j, P_h\}$.

Set $n' \leftarrow n - |\text{BAD}|$ and $t' \leftarrow t - |\text{BAD}|/2$, and return to Step 1.

Theorem 3. *Consider an execution of the batched agreement protocol $\Pi'_{\text{Agreement}}$ using a total-order broadcast channel over an original party set $\mathcal{P}$ of size n , among which at most t parties are corrupted. Assume that no honest shareholder complains against an honest dealer. Then the protocol eventually terminates,*

and all honest parties derive the same output $(\mathcal{P}', n', t', \text{HOLD}, \text{QUAL})$ with the following properties.

- For some $0 \leq x \leq t$, exactly x disjoint pairs have been removed, each containing at least one corrupted party. Moreover, $\mathcal{P}' \subseteq \mathcal{P}$, $n' = n - 2x$, $t' = t - x$, and $n' - 2t' = n - 2t$.
- The final sets satisfy $|\text{HOLD}| = |\text{QUAL}| = n' - t'$, and all complaint messages selected in the final round are empty.
- Except with probability at most $t \cdot \binom{n-t}{n-2t} (|\mathcal{V}|/|\Gamma|)^{n-2t}$, every honest shareholder in HOLD holds valid shares from every dealer in QUAL .

The proof of Theorem 3 appears in Appendix B.2.

5 DASH3: Fewer Commitments and Lightweight Reconstruction

The optimized construction in Section 4 reduces the cost of checking a batch of nonce shares, but each dealer still publishes $d' + 1$ group elements for every nonce and blinding polynomial. We can further reduce this number to a by publishing commitments only at the packing points. These commitments suffice to obtain the partial signatures used in the signatures, but do not support individual verification of signature shares. Under the stronger bound $n \geq 4t + 2a - 1$, we use an alternative signature aggregation protocol that does not require verifying each signature share individually.

5.1 The Main Protocol

In Protocol Π_{DASH2} , each dealer commits to its nonce and blinding polynomials at $d' + 1$ points. The commitments at the a packing points give the group elements corresponding to the packed secrets. Together with the remaining commitments, they allow an aggregator to interpolate the public nonce evaluations at the shareholders' indices and verify individual signature shares in the online phase.

DASH3 retains only the commitments at the packing points $\mathcal{V} = [1 - a, 0]$, reducing each dealer's commitment message from $(d' + 1)(L + n)$ to $a(L + n)$ group elements. These commitments suffice for the batched checks to establish the consistency needed for nonce generation. Each shareholder performs the same private checks as in DASH2 and restricts the public checks to $\mathcal{V}$. Together with agreement, these checks ensure that, except with their soundness error, all honest shareholders in HOLD hold consistent degree- d' nonce sharings from every dealer in QUAL , with packed values matching that dealer's commitments. The parties can then apply the same extraction step as in DASH2 to obtain shared nonces and their partial signatures.

Protocol Π_{DASH3} in Appendix A.2 specifies the resulting construction under the stronger bound $n \geq 4t + 2a - 1$. Relative to DASH2, Step 1 of ephemeral randomness generation publishes commitments only at the a packing points, and Step 3 restricts the public checks in the agreement protocol to those points. Since

the retained commitments no longer allow an aggregator to verify individual signature shares, signature aggregation uses the two-phase reconstruction described in Section 5.2. The parties partition the requested packed signature sharings into ordered blocks of at most $n' - 2t'$ sharings and invoke Π'_{Online} on each block.

5.2 Online Signing

With reduced nonce commitments, an aggregator cannot verify individual signature shares and must reconstruct the sharing polynomial from potentially faulty values. For notational simplicity, we write $\mathcal{P}' = \{P_1, \dots, P_{n'}\}$ for the active set with corruption bound t' . Agreement with values from $d' + t' + 1$ distinct senders certifies a polynomial of degree at most d' , as at least $d' + 1$ of these values come from honest parties and uniquely determine it. To guarantee termination even when corrupted parties withhold their shares, **HOLD** must contain at least $d' + t' + 1$ honest shareholders, as ensured by the stronger parameter bound:

$$|\text{HOLD} \cap \mathcal{H}| \geq n' - 2t' = n - 2t \geq d' + t + 1 \geq d' + t' + 1,$$

where $\mathcal{H}$ is the honest party set. Public share verification in the previous section reduces this requirement to $d' + 1$ honest shareholders in **HOLD**.

We adapt the two-phase public reconstruction of [32] to packed sharings, processing a block of $B \leq n' - 2t'$ sharings to recover aB signatures.

First stage: reconstructing encoded responses. For $h \in [B]$, let $\mathbf{Y}_h$ be the degree- d' polynomial satisfying $\mathbf{Y}_h(i) = [\sigma_h]_i$ for honest $P_i \in \text{HOLD}$. Each shareholder P_i treats its B shares as values at $1, \dots, B$, computes $[\sigma_j]_i = \sum_{h \in [B]} \lambda_{j,h} [\sigma_h]_i$ for each extension point $j \in [B+1, n']$, and sends the corresponding share $[\sigma_j]_i$ to every $P_j \in \mathcal{P}'$. The values received from honest shareholders allow each active party P_i to reconstruct either the original sharing polynomial $\mathbf{Y}_i$ for $i \in [B]$ or the linear combination $\mathbf{Y}_i = \sum_{h \in [B]} \lambda_{i,h} \mathbf{Y}_h$ for $B < i \leq n'$.

To recover $\mathbf{Y}_i$ by online error correction, the receiver first collects $d' + t' + 1$ values from distinct senders and checks whether the polynomial interpolated through $d' + 1$ of them agrees with the rest. If this check fails, it collects further values and invokes Reed–Solomon decoding until a polynomial of degree at most d' agrees with values from at least $d' + t' + 1$ distinct senders. It then evaluates the recovered polynomial at the packing points to obtain $\sigma_i = (\mathbf{Y}_i(0), \dots, \mathbf{Y}_i(1-a))$.

Second stage: recovering the signatures. Each $P_i \in \mathcal{P}'$ sends σ_i to every party in $\mathcal{P}'$. For each $v \in [a]$, let $\mathbf{P}_v$ be the polynomial of degree at most $B - 1$ satisfying $\mathbf{P}_v(h) = \mathbf{Y}_h(1 - v)$ for $h \in [B]$. By the linear extension in the first stage, $\mathbf{P}_v(i) = \mathbf{Y}_i(1 - v)$ for every $i \in [n']$, so each honest party P_i sends $\mathbf{P}_v(i)$ to every active party as the v -th component of its vector. Each party recovers $\mathbf{P}_v$ using the same online error correction procedure, with B values for interpolation and agreement with at least $B + t'$ distinct senders required for acceptance. Since at least $n' - t'$ active parties are honest, the bound $B \leq n' - 2t'$ guarantees that enough correct vectors eventually arrive. Evaluating $\mathbf{P}_v$ at $h \in [B]$ recovers

the unshifted response $\mathbf{Y}_h(1-v)$, from which the parties obtain the signature $(\Delta + R_{h,v}, \delta + \mathbf{P}_v(h))$ on $M_{h,v}$ by adding the common offset δ .

Protocol 6: Π'_{Online}

Precondition: The protocol is run by the n' active parties in $\mathcal{P}'$, at most t' of whom are corrupted. For notational simplicity, we assume $\mathcal{P}' = \{P_1, \dots, P_{n'}\}$. For a block of $B \leq n' - 2t'$ packed signature sharings $[\sigma_1], \dots, [\sigma_B]$, every $P_i \in \text{HOLD}$ holds $\{[\sigma_h]_i\}_{h \in [B]}$, and the corresponding values δ , Δ , $\{M_{h,v}\}_{h \in [B], v \in [a]}$ and $\{R_{h,v}\}_{h \in [B], v \in [a]}$ are public.

1. For $j \in [B+1, \dots, n']$ and $h \in [B]$, each $P_i \in \text{HOLD}$ computes $[\sigma_j]_i = \sum_{h \in [B]} \lambda_{j,h} [\sigma_h]_i$. It sends $[\sigma_j]_i$ to P_j for $j \in [n']$.
2. Each $P_i \in \mathcal{P}'$ checks whether the first $d' + t' + 1$ received values are consistent with a degree- d' polynomial $\mathbf{Y}_i$. If not, it collects additional values from distinct senders and applies Reed–Solomon decoding until it finds the unique degree- d' polynomial agreeing with at least $d' + t' + 1$ received values. It then computes $\sigma_i = (\mathbf{Y}_i(0), \dots, \mathbf{Y}_i(1-a))$.
3. Each $P_i \in \mathcal{P}'$ sends σ_i to every party in $\mathcal{P}'$. For every $v \in [a]$, each party checks whether the v -th components of the first $B + t'$ received vectors are consistent with a degree- $(B-1)$ polynomial $\mathbf{P}_v$. If not, it collects additional vectors and applies Reed–Solomon decoding until it finds the unique $\mathbf{P}_v$ agreeing with the v -th components of at least $B + t'$ received vectors. The party proceeds after reconstructing $\mathbf{P}_v$ for all $v \in [a]$.
4. For every $h \in [B]$ and $v \in [a]$, each party outputs $(\Delta + R_{h,v}, \delta + \mathbf{P}_v(h))$ as the Schnorr signature on $M_{h,v}$.

6 Analysis

Let λ , λ' and λ'' denote the bit lengths of scalars, compressed group elements and uncompressed group elements, respectively, and let κ denote the output length of the hash functions. For elliptic-curve groups, compressed encodings store one affine coordinate and a recovery bit, whereas uncompressed encodings store both coordinates. These lengths typically satisfy $\lambda'' \approx 2\lambda' \approx 2\lambda$. We assume $\kappa = \mathcal{O}(\lambda)$ and $L \gg n$, with $a = \Theta(n)$ for protocols using packed secret sharing. The complexity analysis excludes the costs of one-time key setup, external ordering and beacon services.

Communication complexity. In Table 2, Encrypted shares and Commitments count the encrypted shares and commitments for the nonce polynomials and the additional blinding polynomials in DASH2 and DASH3, while Response polynomials counts the opened response polynomials. Agreement gives the communication of invoking $\Pi_{\text{Agreement}}$ or $\Pi'_{\text{Agreement}}$, while Nonce extraction gives the communication of invoking Π_{SIM} in all three constructions.

Table 2. Amortized offline communication per presignature in the optimistic case, in bits.

Scheme	Encrypted shares	Commitments	Response polynomials	Agreement	Nonce extraction
DASH1	$\mathcal{O}(n\lambda)$	$\mathcal{O}(n\lambda)$	—	$\mathcal{O}(\frac{\kappa n \log n}{L})$	$\mathcal{O}(n\lambda)$
DASH2/3	$\mathcal{O}(n\lambda)$	$\mathcal{O}(n\lambda)$	$\mathcal{O}(\frac{n^2\lambda}{L})$	$\mathcal{O}(\frac{\kappa n \log n}{L})$	$\mathcal{O}(n\lambda)$

With $b = n - 2t$ and $n \mid L$, each batch yields $ab(n - t)L/n = \Theta(n^2L)$ usable presignatures, over which the communication is amortized. Share distribution, commitments and nonce extraction each contribute $\mathcal{O}(n\lambda)$ bits per presignature, whereas response polynomials and agreement contribute lower order terms under the stated parameter assumptions. In the online phase, all three constructions incur an amortized communication cost of $\mathcal{O}(n\lambda)$ bits per signature.

In the pessimistic case, handling complaints in DASH1 does not change its asymptotic communication cost, so the amortized bound remains $\mathcal{O}(n\lambda)$ per presignature, as in the optimistic case. In DASH2 and DASH3, complaints may cause the agreement protocol to run for additional rounds, increasing the amortized communication bound to $\mathcal{O}(n\lambda \log n)$ under the assumption $L \gg n$. If we strengthen the batch size assumption to $L \gg n \log n$, the additional rounds contribute only a lower order term, so the amortized communication remains $\mathcal{O}(n\lambda)$ per presignature. The detailed communication cost is analysed in Appendix D.1.

Computation complexity. In Table 3, Encrypted shares counts share encryption and decryption, while Commitments counts only the generation of commitments to be broadcast. Response polynomials covers response generation in DASH2 and DASH3, which uses only field operations and hence contributes no scalar/group multiplications. Agreement counts direct share verification in $\Pi_{\text{Agreement}}$ and the batched consistency checks in $\Pi'_{\text{Agreement}}$. Nonce extraction counts both computing the assigned block and checking the received blocks in Π_{SIM} .

Amortized over the same $\Theta(n^2L)$ usable presignatures as in the communication analysis, the offline cost is $\mathcal{O}(1)$ group operations per party per presignature in all three constructions. Commitment generation, share encryption and decryption contribute only lower order terms, while the batched consistency checks in DASH2 and DASH3 replace most full multiplications with tiny ones. During online signing, DASH1 and DASH2 require $\mathcal{O}(n)$ group operations per signature at the aggregator, whereas DASH3 requires $\mathcal{O}(n)$ field operations and $\mathcal{O}(1)$ group operations per signature per party. Section D.2 provides the detailed analysis.

In the pessimistic case, DASH2 and DASH3 reuse their previous check results and check only dealers newly added to QUAL in $\Pi'_{\text{Agreement}}$. After $x \leq t$ pairs have been removed, each party has checked at most $n - t + x \leq n$ distinct dealers

Table 3. Amortized offline computation per presignature per party in the optimistic case. Full and Tiny denote scalar/group multiplications by full-length scalars and short coefficients, respectively.

Scheme	Type	Encrypted shares	Commitments	Response polynomials	Agreement	Nonce extraction
DASH1	Full	$\mathcal{O}(1/n)$	$\mathcal{O}(1/n)$	—	$\mathcal{O}(1)$	$\mathcal{O}(1)$
	Tiny	—	—	—	—	—
DASH2/3	Full	$\mathcal{O}(1/n)$	$\mathcal{O}(1/n)$	—	$\mathcal{O}(1/L)$	$\mathcal{O}(1)$
	Tiny	—	—	—	$\mathcal{O}(1)$	—

across all rounds. The total group work thus remains $\mathcal{O}(n^2L)$ per party and is still amortized over $\Theta(n^2L)$ usable presignatures, so additional agreement rounds do not increase the asymptotic complexity. DASH1 verifies public complaints in a single agreement round, skipping the remaining entries in a complaint message after a failed verification and entries against dealers already proven faulty, at an additional cost of $\mathcal{O}(t(d' + 1))$ scalar/group multiplications per party per batch.

AI Disclosure

This paper was edited for grammar and style using GPT-6 Astra.

References

1. M. Abe and S. Fehr. Adaptively secure feldman vss and applications to universally-composable threshold cryptography. In *CRYPTO 2004*, volume 3152 of *LNCS*, pages 317–334, 2004.
2. R. Bacho, S. Das, J. Loss, and L. Ren. Adaptively secure three-round threshold Schnorr signatures from DDH. In *Advances in Cryptology—CRYPTO 2025, Part VI*, volume 16005 of *Lecture Notes in Computer Science*, pages 390–422. Springer, 2025.
3. R. Bacho, S. Das, J. Loss, and L. Ren. Glacius: Threshold Schnorr signatures from DDH with full adaptive security. In *Advances in Cryptology—EUROCRYPT 2025, Part II*, volume 15602 of *Lecture Notes in Computer Science*, pages 304–334. Springer, 2025.
4. R. Bacho, J. Loss, G. Stern, and B. Wagner. HARTS: High-threshold, adaptively secure, and robust threshold Schnorr signatures. In *Advances in Cryptology—ASIACRYPT 2024, Part III*, volume 15486 of *Lecture Notes in Computer Science*, pages 104–140. Springer, 2024.
5. R. Bacho, J. Loss, G. Stern, and B. Wagner. HARTS: High-threshold, adaptively secure, and robust threshold schnorr signatures. Cryptology ePrint Archive, Paper 2024/280, 2024.

6. Z. Beerliova-Trubiniova and M. Hirt. Perfectly-secure MPC with linear communication complexity. In R. Canetti, editor, *Theory of Cryptography Conference—TCC 2008*, volume 4948 of *Lecture Notes in Computer Science*, pages 213–230. Springer, 2008.
7. M. Bellare, E. C. Crites, C. Komlo, M. Maller, S. Tessaro, and C. Zhu. Better than advertised security for non-interactive threshold signatures. In *Advances in Cryptology—CRYPTO 2022, Part IV*, volume 13510 of *Lecture Notes in Computer Science*, pages 517–550. Springer, 2022.
8. E. Ben-Sasson, D. Carmon, S. Kopparty, and D. Levit. Elliptic curve fast fourier transform (ECFFT) part I: Low-degree extension in time $O(n \log n)$ over all finite fields. In *SODA 2023*, pages 700–737. SIAM, 2023.
9. F. Benhamouda, S. Halevi, H. Krawczyk, Y. Ma, and T. Rabin. SPRINT: High-throughput robust distributed Schnorr signatures. In *Advances in Cryptology—EUROCRYPT 2024, Part V*, volume 14655 of *Lecture Notes in Computer Science*, pages 62–91. Springer, 2024.
10. A. Boldyreva. Threshold signatures, multisignatures and blind signatures based on the Gap-Diffie-Hellman-Group signature scheme. In *Public Key Cryptography—PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 31–46. Springer, 2003.
11. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In *Advances in Cryptology—ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer, 2001.
12. R. Canetti. *Studies in Secure Multiparty Computation and Applications*. PhD thesis, Weizmann Institute of Science, Rehovot, Israel, Mar. 1996. Originally submitted June 1995; revised March 1996.
13. R. Canetti, R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Adaptive security for threshold cryptosystems. In M. J. Wiener, editor, *CRYPTO 1999*, volume 1666 of *Lecture Notes in Computer Science*, pages 98–115. Springer, 1999.
14. H. Chu, P. Gerhart, T. Ruffing, and D. Schröder. Practical Schnorr threshold signatures without the algebraic group model. In *Advances in Cryptology—CRYPTO 2023, Part I*, volume 14081 of *Lecture Notes in Computer Science*, pages 743–773. Springer, 2023.
15. E. C. Crites, J. Katz, C. Komlo, S. Tessaro, and C. Zhu. On the adaptive security of FROST. In *Advances in Cryptology—CRYPTO 2025, Part VI*, volume 16005 of *Lecture Notes in Computer Science*, pages 480–511. Springer, 2025.
16. Y. Desmedt. Society and group oriented cryptography: A new concept. In C. Pomerance, editor, *CRYPTO 1987*, volume 293 of *Lecture Notes in Computer Science*, pages 120–127. Springer, 1987.
17. Y. Desmedt and Y. Frankel. Threshold cryptosystems. In G. Brassard, editor, *CRYPTO 1989*, volume 435 of *Lecture Notes in Computer Science*, pages 307–315. Springer, 1989.
18. C. Ganesh and A. Patra. Optimal extension protocols for byzantine broadcast and agreement. *Distributed Comput.*, 34(1):59–77, 2021.
19. S. Gao. A new algorithm for decoding Reed–Solomon codes. In V. K. Bhargava, H. V. Poor, V. Tarokh, and S. Yoon, editors, *Communications, Information and Network Security*, volume 712 of *The Springer International Series in Engineering and Computer Science*, pages 55–68. Springer, 2003.
20. R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Secure distributed key generation for discrete-log based cryptosystems. *J. Cryptol.*, 20(1):51–83, 2007.

21. P. Gerhart, D. Li Calsi, L. Russo, and D. Schröder. Fully-adaptive two-round threshold Schnorr signatures from DDH. In *Advances in Cryptology—EUROCRYPT 2026, Part II*, volume 16542 of *Lecture Notes in Computer Science*, pages 251–283. Springer, 2026.
22. J. Groth and V. Shoup. On the security of ECDSA with additive key derivation and presignatures. In *Advances in Cryptology—EUROCRYPT 2022*, volume 13275 of *Lecture Notes in Computer Science*, pages 365–396. Springer, 2022.
23. J. Groth and V. Shoup. Fast batched asynchronous distributed key generation. In *Advances in Cryptology—EUROCRYPT 2024, Part V*, volume 14655 of *Lecture Notes in Computer Science*, pages 370–400. Springer, 2024.
24. M. Hirt, U. Maurer, and B. Przydatek. Efficient secure multi-party computation. In T. Okamoto, editor, *Advances in Cryptology—ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 143–161. Springer, 2000.
25. M. Hirt and J. B. Nielsen. Robust multiparty computation with linear communication complexity. In C. Dwork, editor, *CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 463–482. Springer, 2006.
26. C. Komlo and I. Goldberg. FROST: Flexible round-optimized Schnorr threshold signatures. In *Selected Areas in Cryptography—SAC 2020*, volume 12804 of *Lecture Notes in Computer Science*, pages 34–65. Springer, 2021.
27. Y. Lindell and A. Nof. Fast secure multiparty ECDSA with practical distributed key generation and applications to cryptocurrency custody. In D. Lie, M. Mannan, M. Backes, and X. Wang, editors, *ACM CCS 2018*, pages 1837–1854. ACM, 2018.
28. K. Nayak, L. Ren, E. Shi, N. H. Vaidya, and Z. Xiang. Improved extension protocols for byzantine broadcast and agreement. In H. Attiya, editor, *DISC 2020*, volume 179 of *LIPIcs*, pages 28:1–28:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.
29. D. Pointcheval and J. Stern. Provably secure blind signature schemes. In *Advances in Cryptology—ASIACRYPT '96*, volume 1163 of *Lecture Notes in Computer Science*, pages 252–265. Springer, 1996.
30. T. Ruffing, V. Ronge, E. Jin, J. Schneider-Bensch, and D. Schröder. ROAST: Robust asynchronous Schnorr threshold signatures. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pages 2551–2564. ACM, 2022.
31. C. Schnorr. Efficient signature generation by smart cards. *J. Cryptol.*, 4(3):161–174, 1991.
32. V. Shoup. AHAB: Asynchronous, high-throughput, adaptively-secure, batched threshold Schnorr signatures. Cryptology ePrint Archive, Paper 2026/620, 2026.
33. V. Shoup and N. P. Smart. Lightweight asynchronous verifiable secret sharing with optimal resilience. *Journal of Cryptology*, 37(3):27, 2024.
34. D. R. Stinson and R. Strobl. Provably secure distributed schnorr signatures and a (t, n) threshold scheme for implicit certificates. In V. Varadharajan and Y. Mu, editors, *ACISP 2001*, volume 2119 of *Lecture Notes in Computer Science*, pages 417–434. Springer, 2001.

Supplementary Material

A Missing Protocols

A.1 Online Signing

The online phase uses the nonce sharings produced in the offline phase to sign the requested messages. After computing the common offset δ and the challenges, each shareholder in HOLD combines its nonce and key shares to compute the packed signature shares and then broadcasts them. For each $u \in [b], \ell \in T$, these shares are evaluations of $\mathbf{Y}^{(u,\ell)} = \mathbf{H}^{(u,\ell)} + \mathbf{Z}^{(u,\ell)}\mathbf{F}$, whose degree is at most $d' = t + 2a - 2$.

To reconstruct the signatures from the broadcast shares, an aggregator checks each received share against the public nonce commitments and the sender's public key share, discarding any share that fails the check. Since $|\text{HOLD}| = n - t$ and at most t shareholders are corrupted, at least $n - 2t \geq d' + 1$ valid shares eventually arrive. Once $d' + 1$ valid shares from distinct shareholders have been collected, the aggregator directly computes the a values $\mathbf{Y}^{(u,\ell)}(1 - v)$ for $v \in [a]$ by Lagrange interpolation. Adding δ to each recovered response and pairing it with $\Delta + R^{(u,v,\ell)}$ gives the corresponding Schnorr signature.

Share verification and signature aggregation can be performed by anyone with access to the broadcast channel, since these operations use only the broadcast shares and public verification information. Therefore, not all parties need to verify shares in the online phase, and parties that receive the resulting signatures can verify them using the ordinary Schnorr verification algorithm.

Protocol 7: Π_{Online}

Precondition: Every $P_i \in \text{HOLD}$ holds $[\sigma^{(u,\ell)}]_i$ for $u \in [b], \ell \in T$. The corresponding values $\delta, \Delta, \mathbf{vk}_i, M^{(u,v,\ell)}, R^{(u,v,\ell)}$, and $\mathbf{Z}^{(u,\ell)}$ are public. Each $P_i \in \text{HOLD}$ broadcasts its signature shares. For every $u \in [b], \ell \in T$, the parties proceed as follows.

1. Upon receiving $[\sigma^{(u,\ell)}]_i$ from $P_i \in \text{HOLD}$, verify $[\sigma^{(u,\ell)}]_i G = \mathbf{H}^{(u,\ell)}(i)G + \mathbf{Z}^{(u,\ell)}(i)\mathbf{vk}_i$ and discard the share if the check fails.
2. Once $d' + 1$ valid shares from distinct shareholders have been collected, for every $v \in [a]$, compute $\mathbf{Y}^{(u,\ell)}(1 - v)$ by Lagrange interpolation and output the Schnorr signature $(\Delta + R^{(u,v,\ell)}, \delta + \mathbf{Y}^{(u,\ell)}(1 - v))$ on $M^{(u,v,\ell)}$.

A.2 The Main Protocol of DASH3

Protocol Π_{DASH3} instantiates the construction of Section 5.1. It follows Protocol Π_{DASH2} , but commits only at the packing points and requires $n \geq 4t + 2a - 1$ to support two-phase reconstruction. Agreement removes disputed pairs while

reusing the distributed nonce shares, and selects the dealers in QUAL for nonce extraction and the shareholders in HOLD for signing.

Protocol 8: Π_{DASH3}

Parameters: Integers $n, t, a \geq 1$ satisfying $n \geq 4t + 2a - 1$, and a batch size $L \geq n$, with degrees $d = t + a - 1$ and $d' = t + 2a - 2$. Let $\mathbb{F}_p$ be a finite field and let $\Gamma \subseteq \mathbb{F}_p$. Denote the party set by $\mathcal{P} = \{P_1, \dots, P_n\}$.

Setup. The step here is the same as in Protocol Π_{DASH1} .

Ephemeral Randomness Generation. Each party $P_i \in \mathcal{P}$ performs the following steps.

1. Choose L random degree- d' nonce polynomials $\{\mathbf{H}_i^{(\ell)}\}_{\ell \in [L]}$ and n random degree- d' blinding polynomials $\{\mathbf{G}_i^{(m)}\}_{m \in [n]}$. For every $v \in [1 - a, 0]$, $\ell \in [L]$ and $m \in [n]$, compute $\hat{\mathbf{H}}_i^{(\ell)}(v) = \mathbf{H}_i^{(\ell)}(v) \cdot G$ and $\hat{\mathbf{G}}_i^{(m)}(v) = \mathbf{G}_i^{(m)}(v) \cdot G$. For every $P_j \in \mathcal{P}$, encrypt under P_j 's public encryption key the shares $\{s_{i,j}^{(\ell)} = \mathbf{H}_i^{(\ell)}(j)\}_{\ell \in [L]}$ and $\{w_{i,j}^{(m)} = \mathbf{G}_i^{(m)}(j)\}_{m \in [n]}$, and broadcast all commitments and ciphertexts. Upon receiving a broadcast message from $P_j \in \mathcal{P}$, P_i initializes Π_{Beacon}^j .
2. Wait for Π_{Beacon}^i to deliver $\{\gamma_i^{(\ell,m)}\}_{\ell \in [L], m \in [n]} \in \Gamma^{L \times n}$. For every $m \in [n]$, compute and broadcast $\mathbf{C}_i^{(m)} = \mathbf{G}_i^{(m)} + \sum_{\ell \in [L]} \gamma_i^{(\ell,m)} \cdot \mathbf{H}_i^{(\ell)}$.
3. Set $\mathcal{V} = [1 - a, 0]$. The parties invoke $\Pi'_{\text{Agreement}}$ on the data generated above to obtain $(\mathcal{P}', n', t', \text{HOLD}, \text{QUAL})$, where $\mathcal{P}'$ is the remaining party set.
4. Let $b = n' - 2t' = n - 2t$ and $c = |\text{QUAL}| = n' - t'$. Let $\Phi \in \mathbb{F}_p^{b \times c}$ be the super-invertible matrix. Order the dealers as $\text{QUAL} = (P_{Q_1}, \dots, P_{Q_c})$ and proceed as follows.
 - For every $v \in [a]$, $\ell \in [L]$ and $j \in [c]$, define the public matrix X by $X_{j,(v,\ell)} = \hat{\mathbf{H}}_{Q_j}^{(\ell)}(1 - v)$. The parties in $\mathcal{P}'$ invoke Π_{SIM} with $(N, f) = (n', t')$ and $\mathcal{V} = [a]$ to obtain retained labels $T \subseteq [L]$ and the partial signatures $R^{(u,v,\ell)} = (\Phi X)_{u,(v,\ell)}$ for each $u \in [b]$, $v \in [a]$ and $\ell \in T$.
 - For every $u \in [b]$ and $\ell \in [L]$, define $\mathbf{H}^{(u,\ell)} = \sum_{j \in [c]} \Phi_{u,j} \mathbf{H}_{Q_j}^{(\ell)}$ and $\mathbf{r}^{(u,\ell)} = (\mathbf{H}^{(u,\ell)}(1 - v))_{v \in [a]}$. For each $u \in [b]$ and $\ell \in T$, every $P_i \in \text{HOLD}$ computes $[\mathbf{r}^{(u,\ell)}]_i = \sum_{j \in [c]} \Phi_{u,j} s_{Q_j,i}^{(\ell)}$.

Signature Share Generation. The step here is the same as in Protocol Π_{DASH1} , using the retained labels T and the sets HOLD and QUAL returned above.

Signature Aggregation. Partition the requested packed signature sharings into ordered blocks $\mathcal{B} = ((u_h, \ell_h))_{h \in [B]}$ with $B \leq b = n' - 2t'$. Within each block, set $[\sigma_h]_i = [\sigma^{(u_h, \ell_h)}]_i$ for every $P_i \in \text{HOLD}$ and $(M_{h,v}, R_{h,v}) = (M^{(u_h, v, \ell_h)}, R^{(u_h, v, \ell_h)})$ for every $v \in [a]$. The parties in $\mathcal{P}'$

invoke Π'_{Online} on each block with corruption bound t' and the common offset (δ, Δ) to output the corresponding Schnorr signatures.

B Proofs of the Agreement Protocols

This section proves the guarantees of the two agreement protocols used by our constructions. The proof of the base protocol relies on publicly verifiable complaints, whereas the batched protocol uses player elimination, since a complaint there only reveals a dispute and does not publicly determine which party is faulty.

B.1 The Base Agreement Protocol

Theorem 1. *Consider an execution of the base agreement protocol $\Pi_{\text{Agreement}}$ by n parties over a total-order broadcast channel. At most t parties are corrupted, and a is the packing parameter. Assume that no verifiable complaint is raised against an honest dealer. Every honest shareholder eventually terminates and outputs the same sets **HOLD**, **QUAL**, and **BAD**. These sets satisfy $|\text{HOLD}| = n - t$ and $|\text{QUAL}| + |\text{BAD}| = n - t$ and have the following properties.*

- No shareholder in **HOLD** complained against any dealer in **QUAL**.
- Every dealer in **BAD** is corrupted and was named in a verifiable complaint from a shareholder in **HOLD**.

*In particular, every honest shareholder in **HOLD** holds valid shares from every dealer in **QUAL**.*

Proof. We prove agreement, the stated properties of the output sets, the validity of the retained shares, and termination.

For agreement, prefix consistency ensures that all honest parties see the same message order. Thus, they select the same first $n - t$ well-formed dealer messages and the same first $n - t$ valid complaint messages. Since all parties use the same public rules to verify complaints, they make the same decisions and obtain the same sets **HOLD**, **QUAL**, and **BAD**.

For the properties of the output sets, the protocol places exactly $n - t$ dealers in the initial set **QUAL** and selects exactly $n - t$ shareholders for **HOLD**. Each dealer removed from **QUAL** is added to **BAD**, so $|\text{QUAL}| + |\text{BAD}| = n - t$ remains true. A dealer is moved to **BAD** only after a complaint from a shareholder in **HOLD** has been verified. By assumption, no such complaint is raised against an honest dealer. Therefore, every dealer in **BAD** is corrupted. Every dealer that is the subject of a verified complaint from **HOLD** is removed, so no such dealer remains in **QUAL**.

In particular, fix an honest shareholder $P_i \in \text{HOLD}$ and a dealer $P_j \in \text{QUAL}$. Party P_i checks every share received from P_j . If any check fails, P_i broadcasts a complaint containing the first failing index, the decrypted share, and a valid

proof of correct decryption, allowing every party to verify the complaint. The protocol then moves P_j to BAD. Since P_j remains in QUAL, every check by P_i must pass. Therefore, every honest shareholder in HOLD holds valid shares from every dealer in QUAL.

For termination, at least $n - t$ parties are honest, and every honest dealer broadcasts a well-formed dealer message. Complete delivery ensures that every honest party eventually receives at least $n - t$ such messages and can select the initial set QUAL. Each honest shareholder then checks all dealers in this set and broadcasts one valid complaint message, which may be empty. Complete delivery again ensures that every honest party eventually receives at least $n - t$ valid complaint messages. Each honest party can then select HOLD, process these messages, and output. $\square$

B.2 The Batched Agreement Protocol

Theorem 3. *Consider an execution of the batched agreement protocol $\Pi'_{\text{Agreement}}$ using a total-order broadcast channel over an original party set $\mathcal{P}$ of size n , among which at most t parties are corrupted. Assume that no honest shareholder complains against an honest dealer. Then the protocol eventually terminates, and all honest parties derive the same output $(\mathcal{P}', n', t', \text{HOLD}, \text{QUAL})$ with the following properties.*

- For some $0 \leq x \leq t$, exactly x disjoint pairs have been removed, each containing at least one corrupted party. Moreover, $\mathcal{P}' \subseteq \mathcal{P}$, $n' = n - 2x$, $t' = t - x$, and $n' - 2t' = n - 2t$.
- The final sets satisfy $|\text{HOLD}| = |\text{QUAL}| = n' - t'$, and all complaint messages selected in the final round are empty.
- Except with probability at most $t \cdot \binom{n-t}{n-2t} (|\mathcal{V}|/|\Gamma|)^{n-2t}$, every honest shareholder in HOLD holds valid shares from every dealer in QUAL.

Proof. We use $\mathcal{P}$ for the original party set and $\mathcal{P}'$ for the updated active set. Let x denote the total number of pairs removed so far.

We first show that all honest parties derive the same output when the protocol eventually terminates. Suppose their states agree at the start of a round. Prefix consistency of the total-order broadcast channel makes them retain and extend QUAL identically and select the same set HOLD. They either all terminate or process the same non-empty complaints in the same order. For each active sender P_j , the current set A_j and the rule of choosing the party with the smallest index in that set determine the same pair to remove. Thus, all honest parties obtain the same updated $\mathcal{P}'$, BAD, n' and t' . Induction over the rounds proves agreement on the output $(\mathcal{P}', n', t', \text{HOLD}, \text{QUAL})$ whenever the protocol terminates.

Each removed pair consists of two distinct active parties and contains at least one corrupted party, since an honest shareholder never complains against an honest dealer. The pairs are disjoint because removed parties never re-enter $\mathcal{P}'$. After x pairs have been removed, $0 \leq x \leq t$, $\mathcal{P}' = \mathcal{P} \setminus \text{BAD}$, $n' = n - 2x$, and $t' = t - x$ bounds the number of remaining corrupted parties. Hence $n' - 2t' = n - 2t$, and at least $(n - t) - x = n' - t'$ honest parties remain active.

For termination, assume that x_i pairs are removed in the i -th round. Removing x_i pairs removes at least x_i dealers from **QUAL** before the next round while decreasing its target size by exactly x_i , so the retained set can be extended to size $n' - t'$. The remaining honest parties supply enough response and complaint messages to complete each round. Their beacon instances complete because all honest parties, including removed ones, continue to initialize instances for dealer messages they receive. Every non-terminating round removes at least one corrupted party, and there are at most t such rounds.

On termination, $|\text{HOLD}| = |\text{QUAL}| = n' - t'$ and all selected complaints are empty by construction. The checks performed by each honest shareholder in **HOLD** have therefore passed for every dealer in **QUAL**. Previously passed checks remain sufficient because the batch data and challenges are unchanged. Except with the soundness error of the batched checks, every such shareholder holds valid shares from every qualified dealer.

We bound this error following [23]. Fix a corrupted dealer P_i , its encrypted shares and commitments are fixed before the beacon output is revealed, after which it broadcasts the response polynomials. If P_i belongs to **QUAL** at termination, every honest shareholder in **HOLD** approves its dealing. A corrupted party in **HOLD** may send an empty complaint even if a check fails, so only honest parties are guaranteed to have passed the checks. There are at least $|\text{HOLD}| - t' = n - 2t \geq d' + 1$ such parties. Interpolate $\mathbf{H}_i^{(\ell)}$ and $\mathbf{G}_i^{(m)}$ from their shares at a fixed $d' + 1$ of the chosen parties, using degree at most d' . If all chosen parties pass their checks, the scalar checks force $\mathbf{C}_i^{(m)} = \mathbf{G}_i^{(m)} + \sum_{\ell \in [L]} \gamma_i^{(\ell, m)} \mathbf{H}_i^{(\ell)}$ for every P_m among them.

If a nonce share at one of these parties is inconsistent with the interpolated polynomial, the scalar check at that party imposes a non-trivial affine equation on the challenge coefficients for each such P_m , which holds with probability at most $1/|\Gamma|$. Otherwise, an incorrect nonce commitment at $v \in \mathcal{V}$ requires

$$\sum_{\ell \in [L]} \gamma_i^{(\ell, m)} (\hat{\mathbf{H}}_i^{(\ell)}(v) - \mathbf{H}_i^{(\ell)}(v)G) = \mathbf{G}_i^{(m)}(v)G - \hat{\mathbf{G}}_i^{(m)}(v).$$

This equation has a non-zero coefficient and holds with probability at most $1/|\Gamma|$. Taking a union bound over the public points, as in the packed variant, gives the conservative bound $|\mathcal{V}|/|\Gamma|$. The coefficients for distinct m are independent, so the probability that all $n - 2t$ chosen parties pass their checks on inconsistent data is at most $(|\mathcal{V}|/|\Gamma|)^{n-2t}$.

Consequently, any inconsistency is witnessed by a $(d' + 1)$ -subset and hence by a subset of exactly $n - 2t$ honest parties. Taking a union bound over the $\binom{n-t}{n-2t}$ possible choices from the original honest parties and at most t corrupted dealers, the probability that an honest shareholder in the final **HOLD** retains inconsistent nonce shares or commitments from a qualified dealer is at most

$$t \binom{n-t}{n-2t} \left(\frac{|\mathcal{V}|}{|\Gamma|} \right)^{n-2t}.$$

With $|\Gamma| = 2^\eta$, choose η so that this bound is negligible. $\square$

Following [23], we choose a typical statistical security parameter of 80. For DASH2 with $n = 49$ and $a = t = 10$, choosing $\eta = 9$ makes the above error bound less than 2^{-80} per agreement invocation, while a full scalar typically has $\lambda = 256$ bits.

C Security Proof

C.1 Distributed Extraction

Theorem 2. *Consider an execution of Π_{SIM} by N computation parties over the total-order broadcast channel. At most f parties are statically corrupted, where $0 \leq f < N/2$. Except with probability at most $f \cdot (N - f)/p$, the following statements hold simultaneously.*

- For a super-invertible matrix Φ , any column of ΦX is uniform over $\mathbb{G}^b$ if the corresponding column of X contains b jointly uniform entries independent of the remaining entries.
- All honest parties select the same $N - f$ accepted blocks and output the same set $T \subseteq \mathcal{L}$ and the same matrix of group elements whose rows are indexed by $[b]$ and whose columns are indexed by $\mathcal{V} \times T$.
- For every $u \in [b]$ and $(v, \ell) \in \mathcal{V} \times T$, the common output equals $(\Phi X)_{u, (v, \ell)}$.

Proof. Since QUAL contains at least $b = N - 2f$ honest parties, each fixed column of X contains at least b honest contributions. If b such entries are jointly uniform over $\mathbb{G}^b$ and independent of the remaining entries, applying the fixed super-invertible matrix Φ yields a column uniformly distributed over $\mathbb{G}^b$.

Let $\mathcal{E}$ be the event that an honest party accepts an incorrect block. Fix an incorrect claimed block and an honest verifier, and choose a column in which the claim differs from the correct product. Write the difference between the claimed and correct columns as $(e_1 G, \dots, e_b G)$, where $\mathbf{e} \neq \mathbf{0}$. Acceptance requires $\boldsymbol{\lambda}^{(i,j)} \mathbf{e}^\top = 0$. The claim is fixed before the verifier samples $\boldsymbol{\lambda}^{(i,j)} \xleftarrow{\$} \mathbb{F}_p^b$, so this equation holds with probability $1/p$. If $f^* \leq f$ parties are corrupted, a union bound over their blocks and the $N - f^*$ honest verifiers gives

$$\Pr[\mathcal{E}] \leq \frac{f^*(N - f^*)}{p} \leq \frac{f(N - f)}{p},$$

where the last inequality follows from $f < N/2$.

Conditioned on $\neg \mathcal{E}$, every honest party rejects every incorrect block, while every correct block passes by linearity. Thus, all honest parties make the same acceptance decisions. Prefix consistency ensures that they select the same first $N - f$ accepted blocks, so they obtain the same set T and the same values. Since every accepted block is correct, the common values are the corresponding entries of ΦX , which proves correctness.

For termination, at least $N - f$ honest computation parties broadcast correct blocks, and every honest party eventually receives and accepts these blocks and therefore terminates. This argument does not depend on $\neg \mathcal{E}$, so termination holds under the stated communication assumptions even if a verification error occurs. $\square$

C.2 Proof of the Main Security Theorem

Assumption 1 (The Discrete Logarithm Assumption). Let λ be the security parameter and let $\mathbb{G}$ be a cyclic group of prime order p with generator G . Consider the following experiment with a discrete logarithm adversary $\mathcal{A}$:

- The challenger and $\mathcal{A}$ receive the description of $\mathbb{G}$, including its order p and generator G .
- The challenger samples $s \xleftarrow{\$} \mathbb{F}_p$ and sends $s \cdot G$ to $\mathcal{A}$.
- The adversary outputs $\bar{s} \in \mathbb{F}_p$.

The adversary wins if $\bar{s} = s$. Write $\mathcal{E}_{\text{DL}}(\mathcal{A}, \mathbb{G})$ for its winning probability. The discrete logarithm assumption states that this probability is negligible in λ for every probabilistic polynomial time adversary $\mathcal{A}$.

Lemma 1 (Union bound for random oracle queries). Let $\mathbb{G}$ have prime order p and generator G . Let q be the number of queries of the form (vk, R, M) that $\mathcal{A}$ made to the random oracle before δ is computed. Let $ab|T|$ be the number of messages to be signed, and denote by $R^{(u,v,\ell)}$, for $u \in [b], v \in [a], \ell \in T$, the group elements that are included in the query where δ is computed. Then the probability that $R^{(u,v,\ell)} + \delta \cdot G$ is included in any of $\mathcal{A}$'s queries before δ is computed is at most $ab|T|q/p$.

Proof. Let $\mathcal{R} \subseteq \mathbb{G}$ be the set of partial signatures in the earlier challenge queries, so that $|\mathcal{R}| \leq q$. For any $X \in \mathbb{G}$, define $\mathcal{R} - X = \{R - X : R \in \mathcal{R}\}$. Then we have $|\mathcal{R} - X| \leq q$ for every $X \in \mathbb{G}$, applying a union bound gives

$$\begin{aligned} & \Pr \left[\exists u \in [b], v \in [a], \ell \in T : R^{(u,v,\ell)} + \delta G \in \mathcal{R} \right] \\ &= \Pr \left[\exists u \in [b], v \in [a], \ell \in T : \delta G \in \mathcal{R} - R^{(u,v,\ell)} \right] \\ &\leq \sum_{u \in [b], v \in [a], \ell \in T} \Pr \left[\delta G \in \mathcal{R} - R^{(u,v,\ell)} \right] \leq \frac{ab|T|q}{p}. \end{aligned}$$

□

Theorem 4. Under Assumption 1, the encryption and complaint assumptions in Section 2.2, and, for DASH2 and DASH3, the beacon simulation assumption specified below, DASH1 and DASH2 with $n \geq 3t + 2a - 1$, and DASH3 with $n \geq 4t + 2a - 1$, are secure threshold signature protocols for generating a single batch of $ab|T|$ signatures in the random-oracle model against a static malicious adversary corrupting at most t parties.

Proof. Agreement and correct reconstruction follow from the preceding results, so it remains to prove unforgeability on a message outside the signing batch. Let $\mathcal{C}$ and $\mathcal{H}$ be the corrupted and honest index sets. Without loss of generality, $|\mathcal{C}| = t$. Given $\text{vk} = s \cdot G$, the simulator uses a successful forger $\mathcal{A}$ to recover s . We follow the simulation of [9, Appendices G.2.1 and G.6], with the additional batch labels and messages required by DASH. It simulates the following parts of the execution:

- The public verification key shares $\mathbf{vk}_j = \mathbf{F}(j) \cdot G$ for the degree- d packed key polynomial $\mathbf{F}$, where $\mathbf{F}(1-v) = s$ for all $v \in [a]$.
- The key shares $[s]_i = \mathbf{F}(i)$ for every $P_i \in \mathcal{C}$.
- The commitments $\hat{\mathbf{H}}_i^{(\ell)}$ to the nonce polynomials $\mathbf{H}_i^{(\ell)}$ of degree d' , for all $i \in \mathcal{H}, \ell \in [L]$.
- The (encryption of the) share $\mathbf{H}_i^{(\ell)}(j)$ for every $i \in \mathcal{H}, j \in \mathcal{C}, \ell \in [L]$.
- The blinding commitments and ciphertexts, beacon transcripts and opened response polynomials related to honest parties in DASH2 and DASH3.
- The agreement messages that decide the sets QUAL, BAD and HOLD, together with the distributed computation messages of Protocol Π_{SIM} .
- The degree- d' response polynomials $\mathbf{Y}^{(u,\ell)}$ for $u \in [b]$ and $\ell \in T$, their honest signature shares and the additional reconstruction messages in DASH3.
- The answers to both the common offset δ and Schnorr challenge queries made by $\mathcal{A}$.

The simulator generates the honest parties' encryption key pairs and retains their decryption keys. It then samples independent uniform shares $\sigma_i \xleftarrow{\$} \mathbb{F}_p$ for $i \in \mathcal{C}$ and gives them to $\mathcal{A}$. For each $z \in [1-a, 0] \cup \mathcal{C}$, let $\mathbf{I}_z$ be the unique polynomial of degree at most $d = t + a - 1$ satisfying $\mathbf{I}_z(z) = 1$ and $\mathbf{I}_z(y) = 0$ for every $y \in [1-a, 0] \cup \mathcal{C}$ with $y \neq z$. Define $\mathbf{I}^* = \sum_{z \in [1-a, 0]} \mathbf{I}_z$ and $\mathbf{F}_{\mathcal{C}} = \sum_{z \in \mathcal{C}} \sigma_z \mathbf{I}_z$. Then $\mathbf{F} = s\mathbf{I}^* + \mathbf{F}_{\mathcal{C}}$ has the required packed values and corrupted shares. Since $\mathbf{I}^*$ and $\mathbf{F}_{\mathcal{C}}$ are known, the simulator publishes $\mathbf{vk}_i = \mathbf{F}_{\mathcal{C}}(i)G + \mathbf{I}^*(i)\mathbf{vk}$ for every $i \in [n]$.

For every $i \in \mathcal{H}$ and $\ell \in [L]$, it samples independent uniform polynomials $\mathbf{A}_i^{(\ell)}$ and $\mathbf{B}_i^{(\ell)}$ of degrees at most $d' = t + 2a - 2$ and $a - 1$, respectively, and represents the nonce polynomial as $\mathbf{H}_i^{(\ell)} = \mathbf{A}_i^{(\ell)} - s\mathbf{B}_i^{(\ell)}\mathbf{I}^*$. At each commitment point v required by the construction, it publishes $\hat{\mathbf{H}}_i^{(\ell)}(v) = \mathbf{A}_i^{(\ell)}(v)G - \mathbf{B}_i^{(\ell)}(v)\mathbf{I}^*(v)\mathbf{vk}$. For $j \in \mathcal{C}$, the share is the known scalar $\mathbf{H}_i^{(\ell)}(j) = \mathbf{A}_i^{(\ell)}(j)$ because $\mathbf{I}^*(j) = 0$. Uniform $\mathbf{A}_i^{(\ell)}$ makes the nonce polynomial uniform and independent of $\mathbf{B}_i^{(\ell)}$.

For DASH2 and DASH3, we assume that the beacon simulator can privately sample honest dealers' coefficients before forming their dealings and reveal them at the prescribed stage, with an adversarial view indistinguishable from a real execution. The simulator samples independent uniform polynomials $\mathbf{C}_i^{(m)}$ of degree at most d' and sets $\mathbf{G}_i^{(m)} = \mathbf{C}_i^{(m)} - \sum_{\ell \in [L]} \gamma_i^{(\ell, m)} \mathbf{H}_i^{(\ell)}$, opening $\mathbf{C}_i^{(m)}$ after the beacon delivers its coefficients. For corrupted dealers, the coefficients are sampled only after their complete dealings have been fixed. The simulator forms the blinding commitments from this identity and encrypts the actual shares addressed to corrupted recipients, where both $\mathbf{H}_i^{(\ell)}(j)$ and $\mathbf{G}_i^{(m)}(j)$ are known. Thus their private checks and all public response checks hold.

Let q bound the number of queries to either random oracle, including the offset query made for signing. The simulator chooses an index uniformly from $[q]$, guessing which fresh offset query binds the signing batch. At that query it reads the ordered dealer set $\text{QUAL}^* = (P_{Q_1}, \dots, P_{Q_c})$, labels T^* and indexed pairs $(R_*^{(u,v,\ell)}, M_*^{(u,v,\ell)})$, answers with $\delta \xleftarrow{\$} \mathbb{F}_p$ and sets $\Delta = \delta G$. It aborts if the

proposed set has fewer than b honest dealers, if two distinct indexed positions contain the same pair (R_*, M_*) , or if a shifted challenge input has already been queried. Conditional on the query and the preceding transcript, the last event has probability at most $ab|T^*|q/p$ by Lemma 1.

The guessed ordering determines Φ . For each $u \in [b]$ and $\ell \in T^*$, set

$$\mathbf{Z}^{(u,\ell)} = \sum_{\substack{j \in [c] \\ Q_j \in \mathcal{H}}} \Phi_{u,j} \mathbf{B}_{Q_j}^{(\ell)}.$$

First suppose that the guessed pairs are distinct. The simulator answers $(\mathbf{vk}, \Delta + R_*^{(u,v,\ell)}, M_*^{(u,v,\ell)})$ with $\mathbf{Z}^{(u,\ell)}(1 - v)$. At least b honest dealers contribute independent hidden polynomials and Φ is super-invertible. Thus the $\mathbf{Z}^{(u,\ell)}$ are independent uniform degree- $(a - 1)$ polynomials, whose evaluations at the a packing points are valid random oracle answers. Every other fresh query receives an independent uniform scalar, and repeated queries retain their answers.

Next, the simulator runs the agreement protocol on behalf of honest parties and treats them as honest dealers even if they broadcast garbage ciphertexts for each other, generates complaints using the recovered plaintexts and computes the honest messages of Π_{SIM} from public group inputs. Once agreement, extraction and the signing request fix the actual tuple, it aborts before releasing signature shares if that tuple differs from its guess. The agreement guarantees imply that QUAL contains at least $b = n - 2t$ honest dealers. Hence the actual tuple is eligible, and the guess matches with probability at least $1/q$, apart from the negligible errors above. Hence, the simulator moves to the next step with probability at least $\frac{1}{q}(1 - \frac{ab|T^*|q}{p})$.

If the simulator has not aborted, its guess matches the actual tuple. For each $u \in [b]$ and $\ell \in T$, substituting the honest dealers' nonce representations gives

$$\begin{aligned} \mathbf{H}^{(u,\ell)} &= \sum_{j \in [c]} \Phi_{u,j} \mathbf{H}_{Q_j}^{(\ell)} \\ &= \sum_{j \in [c], Q_j \in \mathcal{C}} \Phi_{u,j} \mathbf{H}_{Q_j}^{(\ell)} + \sum_{j \in [c], Q_j \in \mathcal{H}} \Phi_{u,j} \mathbf{H}_{Q_j}^{(\ell)} \\ &= \sum_{j \in [c], Q_j \in \mathcal{C}} \Phi_{u,j} \mathbf{H}_{Q_j}^{(\ell)} + \sum_{j \in [c], Q_j \in \mathcal{H}} \Phi_{u,j} \mathbf{A}_{Q_j}^{(\ell)} - s \left(\sum_{j \in [c], Q_j \in \mathcal{H}} \Phi_{u,j} \mathbf{B}_{Q_j}^{(\ell)} \right) \mathbf{I}^*, \end{aligned}$$

Define $\mathbf{H}_{\text{clr}}^{(u,\ell)} = \sum_{j \in [c], Q_j \in \mathcal{C}} \Phi_{u,j} \mathbf{H}_{Q_j}^{(\ell)} + \sum_{j \in [c], Q_j \in \mathcal{H}} \Phi_{u,j} \mathbf{A}_{Q_j}^{(\ell)}$, which is known once the corrupted dealers' polynomials have been recovered. The corresponding signature response polynomial satisfies

$$\begin{aligned} \mathbf{Y}^{(u,\ell)} &= \mathbf{H}^{(u,\ell)} + \mathbf{Z}^{(u,\ell)} \cdot \mathbf{F} \\ &= \mathbf{H}_{\text{clr}}^{(u,\ell)} - s \left(\sum_{j \in [c], Q_j \in \mathcal{H}} \Phi_{u,j} \mathbf{B}_{Q_j}^{(\ell)} \right) \mathbf{I}^* + \mathbf{Z}^{(u,\ell)} (\mathbf{F}_{\mathcal{C}} + s \mathbf{I}^*) \\ &= \mathbf{H}_{\text{clr}}^{(u,\ell)} + \mathbf{Z}^{(u,\ell)} \mathbf{F}_{\mathcal{C}}. \end{aligned}$$

The unknown key term cancels, so the simulator can compute the entire degree- d' response polynomial, send $\mathbf{Y}^{(u,\ell)}(i)$ for each honest signing shareholder and generate the additional linear combinations used in DASH3. Reconstruction outputs $(R^{(u,v,\ell)} + \Delta, \mathbf{Y}^{(u,\ell)}(1 - v) + \delta)$, as required.

Finally, applying the forking lemma [29, Theorem 10] to the complete simulation gives two valid responses with the same (vk, R^*, M^*) and distinct challenges e, e' . From $\sigma \cdot G = R^* + e \cdot \text{vk}$ and $\sigma' \cdot G = R^* + e' \cdot \text{vk}$, the simulator recovers $s = (\sigma - \sigma')(e - e')^{-1}$, contradicting the discrete logarithm assumption. $\square$

D Detailed Complexity Analysis

In this section, we analyze the communication and computational costs of our constructions for the $n \geq 3t + 2a - 1$ and $n \geq 4t + 2a - 1$ settings. We amortize communication costs over the usable presignatures produced by each batch and report computational costs per signature per party, excluding the one-time key setup.

D.1 Communication Complexity

Following the standard convention for maliciously secure protocols, we only count communication sent by honest parties. The arbitrary messages from corrupted parties are excluded, as otherwise the adversary could inflate the measured communication without bound.

We measure communication complexity as the total number of bits sent by honest parties to all parties. We instantiate reliable broadcast with the compact broadcast protocol of [33], which combines erasure coding with Merkle trees and requires only collision-resistant hashing as a cryptographic primitive. Broadcasting an $|m|$ -bit message costs

$$\frac{n(n+1)}{n-2t}|m| + \mathcal{O}(\kappa n^2 \log n)$$

bits in the asynchronous setting, where κ is the output length of the hash function. We write $\rho = n(n+1)/(n-2t)$ for the payload expansion factor of compact broadcast. When $a = t$ and n is chosen at the lower bound in each setting, $\rho \approx 5n/3$ for $n = 3t + 2a - 1$ and $\rho \approx 3n/2$ for $n = 4t + 2a - 1$.

D.1.1 DASH1

Key setup. The long-term signing key is generated only once and reused across many batches, so its communication cost is excluded from the amortized cost per signature.

Presignature generation. At the end of the agreement protocol, we have $|\text{QUAL}| + |\text{BAD}| = n - t$ and $b = n - 2t$. Assuming $n \mid L$, Π_{SIM} retains $n - t$ label blocks and hence $|T| = (n - t)L/n$. Since each polynomial contains a packing points, the batch produces

$$Y = ab|T| = \frac{ab(n - t)}{n}L$$

usable presignatures. When $a = t$ and n is chosen at the lower bound in each setting, $Y \approx 12t^2L/5$ for $n = 3t + 2a - 1$ and $Y \approx 10t^2L/3$ for $n = 4t + 2a - 1$.

- When distributing the nonce polynomials, each dealer broadcasts $d' + 1$ group elements and n encrypted shares for every $\ell \in [L]$. We reuse the ElGamal randomness across the n receiver public keys following SPRINT [9], so the encryptions occupy only $n + 1$ compressed group elements. The dealers therefore broadcast at most $n((n + 1)\lambda' + (d' + 1)\lambda'')L$ bits in this step.
- The base agreement protocol $\Pi_{\text{Agreement}}$ broadcasts at most n complaint messages. Each valid complaint contains at most t corrupted dealers, each specifying a dealer index, a failing index ℓ^* , the corresponding decrypted share and a proof of correct decryption, giving a total payload of at most $tn(\log n + \log L + \lambda + |\text{zkp}|)$ bits. In the optimistic case, it costs a single round of empty complaint messages.
- In Π_{SIM} , each party broadcasts $|\mathcal{V}|bL/n$ group elements. Since $|\mathcal{V}| = a$, the parties broadcast abL group elements in total.

Before implementing compact broadcast, the total broadcast payload is $C = (n(n + 1)\lambda' + n(d' + 1)\lambda'' + ab\lambda'')L$ bits in the optimistic case. Compact broadcast multiplies this payload by ρ and contributes an additional $\mathcal{O}(\kappa n^3 \log n)$ bits per batch. The amortized communication cost is therefore $(\rho C + \mathcal{O}(\kappa n^3 \log n))/Y$ bits per usable presignature. In this setting, choosing $a = t \approx n/5$ and $L \gg \log n$ gives $(\rho C + \mathcal{O}(\kappa n^3 \log n))/Y \approx 1525n\lambda/36$ bits.

Online signing. Each signature share is publicly verifiable against the corresponding commitment and public verification key share, and any $d' + 1$ valid shares uniquely determine the packed signature polynomial. Since one packed share produces a signatures and $|\text{HOLD}| = n - t$, the payload communication is $\rho(n - t)\lambda/a$ bits per signature, excluding the batch level overhead of compact broadcast. When $a = t$ and the lower bound of n is used, this simplifies to exactly $20n\lambda/3 + \mathcal{O}(1)$ bits per signature.

D.1.2 DASH2

The communication analysis for DASH2 follows that in Section D.1.1, with the following differences.

- Each dealer additionally shares n degree- d' blinding polynomials and broadcasts n response polynomials. Across all n dealers, the additional broadcast payload is $n^2((n + 1)\lambda' + (d' + 1)(\lambda'' + \lambda))$ bits per batch. This additional communication is negligible when $L \gg n$.

- In the agreement protocol $\Pi'_{\text{Agreement}}$, each complaint message sent by an honest party contains at most t dealer indices. At most t elimination rounds followed by one final round contribute $\mathcal{O}(nt^2 \log n)$ bits of complaint payload, which is independent of L and amortized over the Y usable presignatures. In the optimistic case, the protocol requires only one round of empty complaint messages.

For $L \gg n$, DASH2 therefore has the same leading amortized communication cost for presignature generation in the optimistic case as DASH1. The online signing procedure is unchanged, so its communication cost is also the same as in Section D.1.1.

D.1.3 DASH3

Here, we focus on the communication costs of ephemeral randomness generation and signature aggregation, while ignoring the setup cost as discussed above.

Presignature generation. We consider the optimistic case, in which no player elimination occurs and agreement terminates after its first complaint round.

- When generating randomness, each dealer distributes L sharing polynomials and n blinding polynomials. For each of the $L + n$ polynomials, the dealer broadcasts a group elements as commitments and encrypts one share for each of the n receivers. Using ElGamal encryption with reused randomness, the n ciphertexts require only $n + 1$ compressed group elements. Therefore, the dealers require broadcasting a total of $n(L + n)((n + 1)\lambda' + a\lambda'')$ bits.
- The correctness and consistency check requires each dealer to broadcast n degree- d' response polynomials, requiring a total of $n^2(d' + 1)\lambda$ bits to be broadcast for the coefficients.
- The agreement step adds at most tn complaint messages, which are empty in the optimistic case because the first complaint round succeeds.
- Π_{SIM} is invoked with $|\mathcal{V}| = a$. Therefore, the total broadcast payload is $abL\lambda''$ bits.

Before implementing compact broadcast, the total optimistic case payload is broadcasting $C = n(L + n)((n + 1)\lambda' + a\lambda'') + n^2(d' + 1)\lambda + abL\lambda''$ bits. Compact broadcast multiplies this payload by ρ and contributes an additional $\mathcal{O}(\kappa n^3 \log n)$ bits per batch. The communication required to realize the public beacon is external to this accounting. The amortized communication cost is therefore $(\rho C + \mathcal{O}(\kappa n^3 \log n))/Y$ bits per usable presignature. When $a = t \approx n/6$ and $L \gg n$, it gives $\rho C/Y \approx 126n\lambda/5$ bits per usable presignature.

Online signing. The signature aggregation protocol processes a block of $B \leq b$ packed signature sharings and outputs aB signatures.

- In the first phase, each party in HOLD sends one share to each of the n parties, which gives at most $n(n - t)\lambda$ bits.

- In the second phase, each party sends its vector of a recovered scalars to every active party, which gives at most $n^2 a \lambda$ bits per block.

The amortized communication payload is therefore at most $\approx n(n - t + na)\lambda/(aB)$ bits per signature. When $B = b$ and $a = t \approx n/6$ at the lower bound on n , the payload term becomes

$$\frac{n(n - t + na)\lambda}{aB} \approx \frac{3}{2}n\lambda + \frac{15}{2}\lambda.$$

D.2 Computation Complexity

We analyze the computation cost per party, amortized over all Y signatures produced by the batch, and count field multiplications and scalar/group multiplications separately.

For reference, evaluating a degree- d' polynomial at m consecutive points from a finite-difference representation requires

$$(m - 1)d' - \frac{d'(d' - 1)}{2}$$

lazy field additions [32]. Initializing or converting this representation costs $d'(d' + 1)/2$ field subtractions per polynomial.

D.2.1 DASH1

Presignature generation. The computation consists of the following parts.

- Each dealer evaluates L degree- d' polynomials at the $n + a$ points in $[-a + 1, n]$, which costs $[(n + a - 1)d' - d'(d' - 1)/2]L$ lazy field additions plus $d'(d' + 1)L/2$ field subtractions if a representation conversion is required.
- Computing the $(d' + 1)L$ commitments costs each dealer the same number of full scalar/group multiplications. Encrypting the nL shares requires an additional $(n + 1)L$ full scalar/group multiplications.
- Each shareholder first decrypts the cL shares received from c selected dealers, with each decryption requiring two full scalar/group multiplications. For one share, the verification costs $d' + 2$ full scalar/group multiplications. Thus, this step performed by each shareholder requires at most $c(d' + 4)L$ full scalar/group multiplications.
- In the optimistic case, all complaint messages are empty, so no complaint verification is required. Complaint handling in $\Pi_{\text{Agreement}}$ can be optimized as follows. Each party skips the remaining entries in a complaint message once an entry fails verification, and skips entries against dealers already proven faulty. Each failed verification ends the processing of a corrupted party's complaint message, while each successful verification identifies a new faulty dealer. Thus, at most $2t$ entries require verification, which requires $\mathcal{O}(t(d' + 1))$ full scalar/group multiplications per party per batch.

- Applying the ECFFT EXTEND algorithm, each assigned block in Π_{SIM} uses $\mathcal{O}(abL \log b/n)$ full scalar/group multiplications when $n \mid L$. Checking up to n claimed blocks costs $\mathcal{O}(nbc)$ field operations to form the check coefficients and $a(b+c)L$ full scalar/group multiplications.
- Each party in HOLD can apply the same ECFFT EXTEND map to its private shares using $\mathcal{O}((n-t)bL \log b/n)$ field operations.

The commitments and encryptions contribute $(d' + n + 2)L$ full scalar/group multiplications, share verification contributes $c(d' + 4)L$ full scalar/group multiplications, and Π_{SIM} contributes $a(b+c)L + \mathcal{O}(abL \log b/n)$ full scalar/group operations. Dividing by Y gives

$$\frac{n[(d' + n + 2) + c(d' + 4) + a(b + c)]}{ab(n - t)} + \mathcal{O}\left(\frac{\log b}{n - t}\right)$$

full scalar/group multiplications per party per usable presignature.

In the optimistic case, we have $c = n - t$. For $a = t \approx n/5$, the amortized number of full scalar/group operations per usable presignature per party is $95/12 + \mathcal{O}(\log n/n)$.

Signature share generation. Each party in HOLD generates one packed signature share for a signatures. Computing the common value $\Delta = \delta G$ requires one full scalar/group multiplication, performed once per signing batch and reused across its packed sharings. Forming the a shifted partial signatures $\Delta + R^{(u,v,\ell)}$ for $v \in [a]$ requires a group additions per party. The remaining challenge computation, interpolation and signature share computation involve only hashing and field operations.

Signature aggregation. An aggregator verifies $d' + 1$ valid shares (after checking at most $d' + t + 1$ shares if up to t of them are invalid) and interpolates the resulting degree- d' polynomial at the a packing points. With straightforward interpolation, this adds $\mathcal{O}(d')$ field operations and $[c(d' + 1 - a) + (d' + 1)(d' + 3)]/a$ full scalar/group multiplications per output signature at the aggregator, in the optimistic case. Fast multipoint interpolation can reduce the field-operation term. For $a = t \approx n/5$, the amortized number of full scalar/group multiplications per signature is $17n/5 + \mathcal{O}(1)$.

D.2.2 DASH2

The computational cost analysis for randomness extraction and online signing in DASH1 also applies to DASH2.

Presignature generation. Batching changes the sharing and verification costs as follows.

- Each dealer evaluates $L + n$ degree- d' polynomials using the method above. Their commitments and encryptions require $(d' + n + 2)(L + n)$ full scalar/group multiplications. Forming the n response polynomials adds $nL(d' + 1)$ tiny scalar/scalar multiplications and the same number of field additions.

- In the optimistic case, each shareholder decrypts $c(L + n)$ shares at a cost of $2c(L + n)$ full scalar/group multiplications. The private response checks require cnL tiny scalar/scalar multiplications and $\mathcal{O}(cn(L + d'))$ other field operations. The public checks use all $d' + 1$ commitment points, costing $c(d' + 1)$ full scalar/group multiplications, $c(d' + 1)L$ tiny scalar/group multiplications and $\mathcal{O}(c(d' + 1)^2)$ field operations to evaluate the responses. Accumulating the public linear combinations also takes $c(d' + 1)L$ group additions.
- The costs of Π_{SIM} and extraction of private shares remain as above. The first complaint round contains only empty messages, so no elimination is needed.

Thus, the amortized number of full scalar/group multiplications per party per usable presignature is

$$\frac{n[(d' + n + 2)(L + n) + c(2L + 2n + d' + 1) + a(b + c)L]}{ab(n - t)L} + \mathcal{O}\left(\frac{\log b}{n - t}\right).$$

The tiny scalar/group multiplications contribute $nc(d' + 1)/(ab(n - t))$ per presignature. For $a = t \approx n/5$ and $L \gg n$, these costs are $35/12 + \mathcal{O}(\log n/n)$ full multiplications and $5 + \mathcal{O}(1/n)$ tiny multiplications.

D.2.3 DASH3

Presignature generation. We consider the optimistic case in which the construction of Section 5.1 requires no player elimination and agreement terminates after its first complaint round. The computation consists of the following parts.

- Each dealer evaluates its L sharing polynomials and n blinding polynomials, all of degree d' , at the n receiver points and the a packing points. This costs $[(n + a - 1)d' - d'(d' - 1)/2](L + n)$ lazy field additions plus $d'(d' + 1)(L + n)/2$ field subtractions if a representation conversion is required.
- Computing the $a(L + n)$ commitments costs each dealer the same number of full scalar/group multiplications. With ElGamal randomness reused across the receiver public keys, encrypting the shares requires an additional $(n + 1)(L + n)$ full scalar/group multiplications.
- For every $m \in [n]$, the dealer P_i forms the coefficient vector of $\mathbf{C}_i^{(m)} = \mathbf{G}_i^{(m)} + \sum_{\ell \in [L]} \gamma_i^{(\ell, m)} \mathbf{H}_i^{(\ell)}$. Across all response polynomials, this costs $nL(d' + 1)$ tiny scalar/scalar multiplications and the same asymptotic number of field additions.
- Each shareholder first decrypts the $c(n + L)$ shares received from c selected dealers, with each decryption requiring two full scalar/group multiplications. Each shareholder verifies the responses from the c selected dealers. Checking the first equation requires cnL tiny scalar/scalar multiplications and $\mathcal{O}(cn(d' + L))$ field operations, while checking the second equation requires ca full scalar/group multiplications, caL tiny scalar/group multiplications and $\mathcal{O}(cad')$ field operations.
- The complaint messages are empty in the optimistic case, so no player elimination is triggered.

- Assuming $n \mid L$, a party computes its assigned block using $\mathcal{O}(abL \log b/n)$ full scalar/group multiplications. Checking up to n claimed blocks costs $\mathcal{O}(nbc)$ field operations to form the check coefficients and $a(b+c)L$ full scalar/group multiplications.
- Each party in HOLD applies the same ECFFT EXTEND map to its private shares using $\mathcal{O}((n-t)bL \log b/n)$ field operations.

Generating the commitments and ciphertexts requires $(n+a+1)(L+n)$ full scalar/group multiplications. Decrypting the shares requires $2c(n+L)$ full scalar/group multiplications, and the public response checks require another ca . Π_{SIM} contributes $a(b+c)L + \mathcal{O}(abL \log b/n)$ full scalar/group multiplications. Dividing by Y gives

$$\frac{n[(n+a+1)(L+n) + c(2n+a+2L) + a(b+c)L]}{ab(n-t)L} + \mathcal{O}\left(\frac{\log b}{n-t}\right)$$

full scalar/group multiplications per party per usable presignature.

In the optimistic case, we have $c = n - t$. For $a = t \approx n/6$ and $L \gg n/\log n$, the amortized computation complexity per party includes $27/10 + \mathcal{O}(\log n/n)$ full scalar/group operations and 1.5 tiny scalar/group operations.

Signature share generation. The analysis here is the same as in the $n \geq 3t + 2a - 1$ setting above.

Signature aggregation. Protocol Π'_{Online} processes a block of $B \leq b$ packed sharings and outputs aB signatures. For a fixed rate Reed–Solomon code of length m and dimension d , let $D_{\text{RS}}(m)$ denote its decoding cost in field operations. Implemented using fast polynomial arithmetic in [19], gives $D_{\text{RS}}(m) = \tilde{\mathcal{O}}(m)$, where $\tilde{\mathcal{O}}$ suppresses polylogarithmic factors.

- In the first step, each party in HOLD forms $(n' - B)$ linear combinations of B shares at a cost of $\mathcal{O}(n'B)$ field operations per block.
- In the optimistic case, the two reconstruction steps add

$$\mathcal{O}(D_{\text{RS}}(d' + 1 + t') + aD_{\text{RS}}(B + t'))$$

field operations per block per party.

- After the first reconstruction, each party evaluates the recovered degree- d' polynomial at a packing points. After the second reconstruction, it evaluates each of the a recovered degree- $(B-1)$ polynomials at B points. These evaluations cost $\mathcal{O}(a(d' + B^2))$ lazy field additions per block per party.
- Forming each partial signature $\Delta + R_{h,v}$ requires one group addition.

With $B = b$ and $a = t \approx n/6$, the amortized per-party computational cost is $\mathcal{O}(n)$ per signature.