Concrete LDVR Attacks with Fewer Signers

Renas Bacho

Ruhr University Bochum, Germany

Abstract. We improve the concrete parameters of the attacks on the low-dimensional vector representation (LDVR) problem by Rodríguez García and Tessaro (ePrint 2026/2126), whose attack on canonical evaluation points uses 5,020 signers. By combining their Chinese remainder theorem construction with a generalized birthday search over many challenges, we obtain a concrete attack with 488 signers and 173 corruptions over the Curve25519 scalar field. In 64 batches of 2^{25} challenges, the search finds solutions for six distinct challenges. We compare the counts measured at each stage of the search with predictions from an explicit heuristic model. For adversarially chosen evaluation points, we build on the binary attacks of Rodríguez García and Tessaro and give a construction that attains the optimal LDVR success probability for every oracle query budget when $2 \leq n \leq 253$ and $t_c = t$ in the uniform oracle model. At 228 signers and 114 corruptions, our implementation finds a successful subset using SHA-512 challenges after a cumulative total of 360.13 million queries across two runs, taking 168.60 seconds in total. We also give an explicit set of 256 evaluation points for which an efficient subset search attack using 128 corruptions succeeds with probability at least 51.92% per uniform challenge. For both canonical and adversarially chosen evaluation points, we analyze the query complexity and report measured and estimated running times.

Keywords: Threshold Schnorr · adaptive security · LDVR · subset sum · generalized birthday attacks

1 Introduction

We consider Shamir-based threshold Schnorr protocols in a group $\mathbb{G} = \langle g \rangle$ of prime order p that expose the public key shares $\mathsf{pk}_i = g^{\mathsf{sk}_i}$, where $\mathsf{sk}_i \in \mathbb{F}_p$ is party i's secret key share. These values may be public from the outset or reconstructible from the adversary's protocol view. This class includes a broad range of threshold Schnorr protocols in the literature, including FROST and Sparkle. We study their security under adaptive corruptions, where the adversary may choose which parties to corrupt throughout the protocol execution based on the information observed so far, including signature challenges. Crites et al. [3] introduce the low-dimensional vector representation (LDVR) problem to capture this attack strategy and show that an efficient LDVR attack yields an adaptive forgery against protocols in this class.

Very recently, Rodríguez García and Tessaro gave a polynomial-time LDVR attack for the canonical evaluation points $x_i = i$ [6]. In this setting, party i holds the share $\mathsf{sk}_i = f(i)$, where $f \in \mathbb{F}_p[X]$ is the degree-t Shamir sharing polynomial satisfying $f(0) = \mathsf{sk}$ for the secret signing key sk. Their optimized concrete attack over the Curve25519 scalar field uses 5,020 signers, corrupts 204 of them, and succeeds with probability approximately 1/194 per query. They also give a binary attack for adversarially chosen evaluation points. For a 256-bit prime modulus, this attack uses 512 signers and 256 corruptions and succeeds with probability one after a single oracle query. For even n with $2^{n/2} < p$ and $t_c = t = n/2$, their construction has per-query success probability $2^{n/2}/p$. This leaves a gap to the general unconditional LDVR bound of Crites et al. [3]: for $t_c = t$, any adversary making at most Q oracle queries satisfies

$$\Pr[\text{LDVR success}] \leq \min\left\{1, \frac{Q}{p}\binom{n}{t}\right\},$$

regardless of its computational power and even when its queries are chosen adaptively. Rodríguez García and Tessaro leave closing this gap as an open question [6].

We study concrete LDVR attacks with the goal of reducing the number of signers required. For both canonical and adversarially chosen evaluation points, we give algorithms that identify sets of parties whose corruption enables an adaptive forgery. We analyze the query complexity of these attacks and report measured and estimated running times.

Our results. For canonical evaluation points, we combine the Chinese remainder theorem construction of Rodríguez García and Tessaro with a generalized birthday search that processes many challenges in a single batch. Our concrete instantiation uses $n = 488$ signers, corrupts $t_c = t = 173$ parties, and has signing threshold $t + 1 = 174$. We first evaluate six preselected batches, each containing 2^{25} simulated LDVR challenges, and find a solution in the fourth batch. Running on one CPU thread, our implementation processes the first four batches in a total of 172.66 seconds. This measurement covers the subset search, while challenge hashing is benchmarked separately. A second experiment covering these six batches and 58 further batches finds solutions for six distinct challenges. Every accepted subset solves its LDVR instance by construction, and a separate verification script confirms this for all of them.

We do not prove a success probability for this search. We instead give an explicit heuristic model for the residue checks and subset sizes and compare its predictions with the counts measured at each stage of the search over the 64 batches. The predicted total number of surviving candidates differs from the measured number by approximately 0.1%. At the final stage, the model predicts that approximately three groups of related candidates pass all residue checks, whereas seven are observed, six of which yield accepted subsets. The correctness of the accepted subsets does not depend on the model. We also optimize the corruption budget and make the reconstruction step more flexible by considering local choices of different sizes, allowing more search candidates to be completed to valid corruption sets of size t.

For adversarially chosen evaluation points, we first strengthen the preceding Q-query upper bound: when $t_c = t$, every adversary making at most Q oracle queries succeeds with probability at most

$$1 - (1 - \rho)^Q, \qquad \rho = \min\left\{1, \frac{1}{p} \cdot \binom{n}{t}\right\}.$$

The bound holds in the uniform oracle model even for computationally unbounded adversaries that choose their queries adaptively. We then refine the binary approach of Rodríguez García and Tessaro and give a family of evaluation points with an efficient subset search algorithm that attains this bound for every $2 \le n \le 253$ and $1 \le t < n$. Thus, the construction is optimal for every oracle-query budget in this parameter range. At $n = 228$ and $t_c = t = 114$, our implementation finds a successful subset after a cumulative total of 360,128,928 SHA-512 queries and 168.60 seconds. A separate verification script confirms that the resulting subset solves the LDVR instance and yields a valid Schnorr verification equation. The bound above gives a budget of approximately 1.46 billion queries for 99% success at these parameters, corresponding to an estimated running time of 10.6–11.6 minutes at our measured query rates. We also quantify how the required query budget grows as the number of signers decreases. Finally, we give a separate explicit construction with 256 evaluation points and $t_c = t = 128$ for which an efficient subset search algorithm succeeds with probability at least 51.92% per uniform challenge.

Experimental artifacts. The accompanying artifact, version 1.0.0, is available on Zenodo [1]. It consists of the core archive `ldvr_experimental_artifacts.zip` and the optional data archive `ldvr_candidate_data.zip`. The core archive contains source code, configurations, experimental records, and verification scripts for our concrete attacks. The data archive contains records of the surviving candidates from all 64 batches of Section 3.5, allowing the measured values in Tables 3 and 4 to be regenerated without rerunning the searches. The documentation provides instructions for verifying the stored solutions and counts and for rerunning the searches and benchmarks.

1.1 Technical overview

From LDVR to reciprocal subset sum. Following Rodríguez García and Tessaro [6], we choose the LDVR query so that the challenge-dependent target vector is $c \cdot \mathbf{e}_0 + \mathbf{e}_1 = (c, 1, 0, \ldots, 0)$, where $c \in \mathbb{F}_p$ is the oracle challenge. Let $\mathbf{v}(x) = (1, x, \ldots, x^t) \in \mathbb{F}_p^{t+1}$ be the evaluation vector associated with $x \in \mathbb{F}_p$. For every subset $S \subseteq [n]$ of size t,

$$c \cdot \mathbf{e}_0 + \mathbf{e}_1 \in \mathrm{Span}\{\mathbf{v}(x_i) : i \in S\} \quad \Longleftrightarrow \quad c = \sum_{i \in S} x_i^{-1}.$$

For the canonical evaluation points $x_i = i$, the problem therefore becomes $\sum_{i \in S} 1/i = c$, $|S| = t$, over $\mathbb{F}_p$. Thus, after receiving the challenge, the adversary must find a size-t subset with the prescribed reciprocal sum. The search itself takes place entirely in the scalar field $\mathbb{F}_p$.

A successful subset directly yields the corruption set needed for an adaptive forgery. Indeed, if $f(X) = a_0 + a_1 X + \cdots + a_t X^t$ is the Shamir sharing polynomial, suitable interpolation coefficients for the selected shares recover $z = a_1 + c a_0$. Consequently, $g^z = R \cdot \mathsf{pk}^c$ for $R = g^{a_1}$ and $\mathsf{pk} = g^{a_0}$, which is the Schnorr verification equation.

Local reconstruction. For canonical evaluation points, we build on the Chinese remainder theorem construction of Rodríguez García and Tessaro. Choose a positive integer L and a collection $\mathcal{Q}$ of small pairwise coprime moduli. The completion indices divide L, while the indices assigned to the local problem modulo $q \in \mathcal{Q}$ divide Lq. All index groups are disjoint. Define $P = \prod_{q \in \mathcal{Q}} q$ and $D = LP$, with $p \nmid D$. Multiplying the reciprocal subset sum equation by D gives

$$\sum_{i \in S} \frac{D}{i} = Dc \pmod{p}.$$

Each index i assigned to the local problem modulo q divides Lq, so its scaled weight $D/i = (P/q)(Lq/i)$ is an integer. This weight vanishes modulo every other modulus $q' \in \mathcal{Q} \setminus \{q\}$, so selecting i affects only the local congruence modulo q. For each q, we therefore precompute subsets of its local indices representing a set of residues modulo q. The completion indices also have integer scaled weights. For the additional search indices, we compute scaled reciprocals directly in $\mathbb{F}_p$.

The indices used for the final completion step divide L. Their scaled weights $D/i = P(L/i)$ are therefore integer multiples of P, so adding these indices does not change any of the local congruences. We use dynamic programming to precompute, for each k, all integer sums $\sum_{i \in B} L/i$ obtainable from a subset B of exactly k completion indices. The resulting table also allows us to recover a subset realizing any stored sum. During reconstruction, if the remaining contribution is Ps and k additional indices are needed to reach size t, we look up whether s is attainable with exactly k indices and, if so, recover a suitable completion.

More precisely, after selecting a partial subset U, let

$$A = \left(Dc - \sum_{i \in U} \frac{D}{i} \right) \bmod p$$

be the least nonnegative representative of the remaining target. When the required local table entries exist, they provide subsets whose scaled weights sum to an integer B satisfying $B \equiv A \pmod{P}$. The completion step must then represent the integer quotient $(A - B)/P$ as a sum of weights L/i using the indices dividing L, while selecting exactly the number of additional indices needed to obtain a size-t subset. We restrict A to an interval that guarantees this completion whenever the required number of indices lies in the permitted range. For our 488-signer parameters, we use 53 completion indices. The dynamic program represents every integer in a common interval of quotient values using exactly k completion indices, for each $18 \le k \le 35$.

The best corruption budget depends on the sizes of the subsets stored in the local tables. For the tables used in our measured attack, $t = 173$ maximizes the fraction of combinations of local choices and partial subsets U that satisfy the final size requirement. Complementing some stored local subsets and adding fixed unused indices also yields size-244 solutions with essentially the same fraction; we do not establish improved search success or running time for this variant. We further extend the local tables to store subsets of several sizes for each residue. A second dynamic program chooses these sizes so that the remaining number of indices falls within the range supported by the completion procedure. This allows more candidates to be completed to valid corruption sets of size t and increases the guaranteed number of pairs (U, A) for which reconstruction succeeds. These counts do not establish a higher attack success probability, since different pairs may correspond to the same challenge.

Searching over many challenges. To apply the local reconstruction procedure, the remaining target must lie in a prescribed interval, and the local residue and subset-size conditions must hold. To find suitable candidates at smaller signer counts, we combine this procedure with a generalized birthday search that processes many challenges in a single batch.

For our measured 488-signer attack, we partition 94 additional indices into seven blocks and enumerate the scaled reciprocal subset sums within each block. Let $\mathcal{I}$ denote the target interval for local reconstruction, and

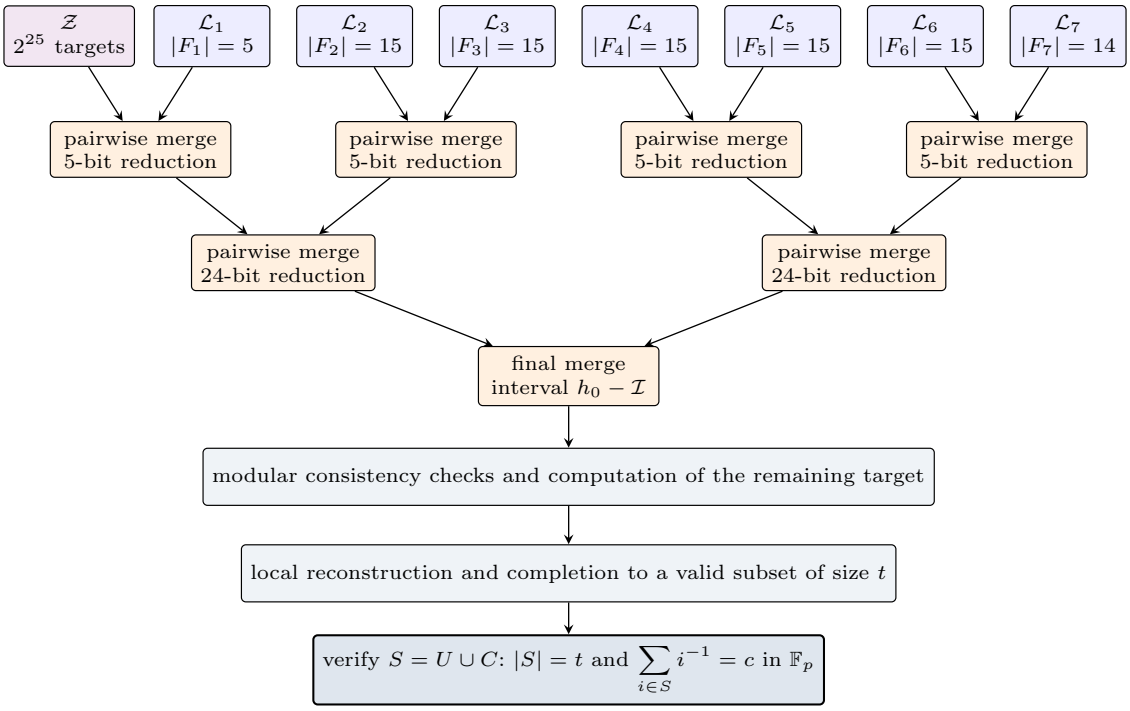


Fig. 1. Eight-list generalized birthday search for the canonical attack. For each $b \in [7]$, the list $\mathcal{L}_b$ contains the scaled reciprocal sums over subsets of F_b and records the selected indices, while $\mathcal{Z} = \{h_0 - Dc_j\}_{j \in [2^{25}]}$ contains the transformed challenge targets and their indices. At each merge level, the algorithm retains sums in a prescribed interval. These intervals become progressively smaller, with the final merge using an enlargement of $h_0 - \mathcal{I}$. Each candidate surviving the final merge identifies a challenge c_j and a partial subset U. The algorithm checks the modular conditions, recomputes the remaining target, and attempts local reconstruction. It accepts only if it obtains a completion C such that $S = U \cup C$ has size t and satisfies $\sum_{i \in S} i^{-1} = c_j$ in $\mathbb{F}_p$.

let h_0 be its midpoint, rounded down. An eighth list contains 2^{25} transformed challenge targets $z_j = h_0 - Dc_j \pmod p$. We then seek one entry from each of the eight lists whose sum lies in $h_0 - \mathcal{I}$, an interval around zero. If U is the union of the seven selected subsets, this sum equals $h_0 - A$ modulo p. Thus, the interval condition ensures that the remaining target A lies in $\mathcal{I}$, after which the local reconstruction procedure checks whether U can be completed to a valid subset of size t.

We process the eight lists in a merge tree following Wagner's generalized birthday algorithm. At each level, we merge pairs of lists and retain sums in a prescribed interval, with the intervals becoming progressively smaller. We track the selected indices and the challenge index throughout the search. At the final merge, modular checks discard pairs whose combined residues are incompatible with the local tables. For each remaining candidate, we recompute the remaining target in $\mathbb{F}_p$ and invoke the local reconstruction procedure. We then verify that the completed subset has size t and satisfies the reciprocal subset sum equation. Processing many challenges together shares the subset sum lists and intermediate merge computations across them.

Every accepted output is a valid LDVR solution. The lists have substantial arithmetic structure, so we do not prove a success probability for the search. Section 3.5 gives an explicit heuristic model for the residue checks and subset-size distribution, derives the counts it predicts at each stage of the search, and compares them with the measured counts.

Adversarially chosen evaluation points. We choose evaluation points whose reciprocals have a binary structure that permits direct recovery of successful subsets. For $2 \le n \le 253$, fix the evaluation points before sampling any oracle challenges by setting $x_i^{-1} = 1 + v_i$, where

$$(v_1, \ldots, v_n) = (0, 2^0, 2^1, \ldots, 2^{n-2}).$$

Adding 1 makes all reciprocals nonzero while preserving their distinctness. For $1 \le t < n$, a size-t subset satisfies the reciprocal subset-sum equation precisely when its selected v_i values sum to $c - t$ modulo p.

Let y be the least nonnegative representative of $(c-t) \bmod p$. If $y \ge 2^{n-1}$, no solution exists. Otherwise, the binary expansion of y uniquely identifies the selected powers of two. These indices form a size-t solution if the Hamming weight of y is t. If the Hamming weight is $t-1$, adding the index corresponding to $v_1 = 0$ gives a size-t solution. Thus, subset recovery requires only a range test, a Hamming weight computation, and recovery of the indices corresponding to the nonzero bits.

Over the Curve25519 scalar field, $2^{n-1} < p$ throughout this parameter range. Consequently, the binary sums do not wrap around modulo p, and distinct size-t subsets yield distinct challenges. The success probability per uniform challenge is therefore exactly $\binom{n}{t}/p$. For Q independent uniform challenges, the success probability is

$$1 - \left(1 - \frac{1}{p} \cdot \binom{n}{t}\right)^Q.$$

This attains our upper bound in the uniform oracle model for every query budget, even against computationally unbounded adversaries whose queries depend on previous oracle answers. Choosing $t = \lfloor n/2 \rfloor$ maximizes the number of successful challenges.

Smaller signer counts reduce the success probability per query while preserving efficient subset recovery. Each query requires only hashing, reduction modulo p, subtraction, a range test, and a Hamming weight test, without a precomputed dictionary. The construction therefore provides a direct tradeoff between the number of signers and the query budget: fewer signers require more independent challenges to achieve a given success probability.

Constant success with 256 *evaluation points.* We extend the preceding binary construction by appending three explicitly specified values a_1, a_2, a_3 to the 253 values $0, 2^0, \ldots, 2^{251}$. For each challenge, the subset search algorithm considers all eight choices of additional indices $J \subseteq \{1, 2, 3\}$ and computes

$$y_J = \left(c - t - \sum_{j \in J} a_j\right) \bmod p.$$

A choice yields a size-t solution if $y_J < 2^{252}$ and its Hamming weight is either $t - |J|$ or $t - |J| - 1$. The binary expansion identifies the selected powers of two, and the index corresponding to $v_1 = 0$ is included in the latter case.

Each choice of J defines a translated set of successful challenges, with the permitted Hamming weights determined by $|J|$. The overall success probability equals the cardinality of the union of these eight sets divided by p. We apply the second-order Bonferroni inequality [4] to bound the union size from below by the sum of the individual set sizes minus the sum of all pairwise intersection sizes. We compute the 28 pairwise intersection sizes using a dynamic program for binary addition, accounting for reduction modulo p. It processes the addition bit by bit, recording the carry passed to the next position and counting the 1 bits in both the original and shifted values.

For the three specified values a_1, a_2, a_3 and $t = 128$, our analysis proves that the success probability is at least 51.92% per uniform challenge. With seven independent uniform challenges, the probability of finding at least one successful subset exceeds 99.4%.

2 Preliminaries

Notation. Write $[n] = \{1, \ldots, n\}$. All vector spaces and inverses are over $\mathbb{F}_p$, unless explicitly taken over the integers. A field element compared to an integer denotes its least nonnegative representative. Throughout the concrete results we use

$$p = 2^{252} + 27742317777372353535851937790883648493, \tag{1}$$

the order of the prime-order subgroup used by Ed25519. The parameter n denotes the number of signers, t is the Shamir polynomial degree, and t_c is the corruption budget. We assume $1 \le t < n$ and $0 \le t_c \le t$. The reconstruction threshold is $t + 1$.

2.1 The LDVR game

Let $x_1, \ldots, x_n \in \mathbb{F}_p^*$ be distinct, set $x_0 = 0$, and define $v(x) = (1, x, \ldots, x^t) \in \mathbb{F}_p^{t+1}$. An adversary learns the evaluation points and queries an oracle on coefficient vectors $\alpha \in \mathbb{F}_p^{n+1}$. For its jth query, the oracle returns an independent uniform scalar c_j. The adversary wins by outputting a query index j and $S \subseteq [n]$ with $|S| \leq t_c$ such that

$$c_j v(0) + \sum_{i=0}^{n} \alpha_j[i] v(x_i) \in \mathrm{Span}\{v(x_i) : i \in S\}. \tag{2}$$

The LDVR assumption asserts that this success probability is negligible for polynomial-time adversaries in the relevant parameter family [3,6]. The oracle returns an independent uniform challenge on every call, even when the coefficient vector is repeated. When challenges are computed using a fixed hash function, we obtain successive challenges by varying the message.

All our attacks set $t_c = t$ and choose α so that $\sum_i \alpha[i] v(x_i) = e_1$, where $e_1 = (0, 1, 0, \ldots, 0)$ and $e_0 = v(0)$. Such coefficients can be computed by interpolation on any $t+1$ points. We therefore study a specific LDVR query whose challenge vector is $c \cdot e_0 + e_1$.

Lemma 1 (Reciprocal query). *For $S \subseteq [n]$ with $|S| = t$, the vector $ce_0 + e_1$ lies in $\mathrm{Span}\{v(x_i) : i \in S\}$ if and only if*

$$c = \sum_{i \in S} x_i^{-1}. \tag{3}$$

For a successful subset, let $Q_S(X) = \prod_{i \in S}(X - x_i)$. The coefficients expressing $ce_0 + e_1$ as a linear combination of the selected evaluation vectors are

$$\lambda_i = -\frac{Q_S(0)}{x_i^2 Q_S'(x_i)}, \qquad i \in S. \tag{4}$$

Proof. The t evaluation vectors are independent, and their span is the kernel of the linear functional whose coefficient vector is Q_S. This functional evaluates $ce_0 + e_1$ to $cQ_S(0) + Q_S'(0)$. Since $Q_S(0) \neq 0$ and $Q_S'(0)/Q_S(0) = -\sum_{i \in S} x_i^{-1}$, its vanishing is equivalent to (3). To obtain (4), divide the value at zero of the Lagrange basis polynomial for x_i by x_i. Interpolation gives the required moments of degrees $1, \ldots, t$, and the preceding identity gives the degree-zero moment. □

From a successful subset to a forgery. Let $f(X) = \sum_{k=0}^{t} a_k X^k$ be the Shamir polynomial, with public key $\mathsf{pk} = g^{a_0}$ and public share commitments $Y_i = g^{f(x_i)}$. Public interpolation computes $R = g^{a_1}$. For distinct messages m_j, query $c_j = H(R, \mathsf{pk}, m_j)$. Once (3) holds, corrupt the selected t parties and compute

$$z = \sum_{i \in S} \lambda_i f(x_i) = a_1 + c_j a_0.$$

Then $g^z = R \cdot \mathsf{pk}^{c_j}$. This attack needs no signing query. It uses the same reduction underlying LDVR [3]; the algebraic search is independent of elliptic-curve coordinates. Corruption is an oracle call in this model, so the cost of compromising real participants is not part of the reported computation.

2.2 Subset sum attacks

The canonical attack of [6] uses the Chinese remainder theorem to combine deterministic algorithms for local subset sum problems. We extend this construction with a generalized birthday search over many challenges. For the reciprocal query, at most $\binom{n}{t}$ challenges admit a solution; this number may be smaller when distinct subsets have the same reciprocal sum. We prove that every accepted output solves LDVR, but do not establish the success probability of the birthday search. Section 3.5 gives an explicit heuristic model and compares its predictions with the measured search counts.

2.3 Unconditional bounds on LDVR success

For small signer counts, LDVR attacks require many oracle queries, regardless of the computation performed between queries. Crites et al. [3] prove the following bound, recalled in [6, Theorem 2.1]: for any distinct nonzero evaluation points, any corruption budget $t_c \le t < n$, and any adversary making at most Q oracle queries,

$$\Pr[\text{LDVR success}] \le \min\left\{1, \frac{Q}{p}\frac{(n)_{2t_c-t}}{(t_c)_{2t_c-t}}\right\}. \tag{5}$$

Here, $(a)_k = a(a-1)\cdots(a-k+1)$ for $k > 0$, and $(a)_k = 1$ for $k \le 0$. The bound allows adaptive queries, arbitrary preprocessing, and unlimited computation; all oracle queries count toward Q. When $t \ge 2t_c$, the ratio equals 1, and the bound reduces to $\min\{1, Q/p\}$.

Counting successful challenges when $t_c = t$. In this case, the ratio in (5) simplifies to $\binom{n}{t}$. To understand this bound, fix a query coefficient vector and a set S of t indices. Since the evaluation points are distinct and nonzero, $v(0)$ and the vectors $\{v(x_i) : i \in S\}$ are linearly independent. Therefore, at most one challenge c allows the query vector to lie in their span. Otherwise, subtracting the equations for two distinct challenges would express $v(0)$ as a linear combination of the selected evaluation vectors.

Any solution using fewer than t indices can be extended to a set of t indices. Taking the union over all $\binom{n}{t}$ such sets therefore shows that, for any fixed query, at most $\binom{n}{t}$ challenges admit a solution. This argument applies to arbitrary query coefficients, including those outside the reciprocal-query construction of (3).

A sharper bound for adaptive queries. Set $\rho = \min\{1, \binom{n}{t}/p\}$. Conditional on the preceding interaction, the adversary chooses its next coefficient vector before receiving the next independent uniform challenge. The counting argument above therefore bounds the conditional probability that this query admits a solution by ρ. This remains true when none of the preceding queries admits a solution.

Consequently, the probability that all Q queries fail to admit a solution is at least $(1-\rho)^Q$. We obtain the sharper bound

$$\Pr[\text{LDVR success}] \le 1 - (1-\rho)^Q. \tag{6}$$

The bound holds even for computationally unbounded adversaries whose queries depend on previous oracle answers. Section 4 gives evaluation points and an efficient algorithm that finds a valid size-t subset whenever one exists. The resulting attack attains the bound for every query budget when $2 \le n \le 253$ and $1 \le t < n$.

Concrete implications. The ratio in (5) is at most $\binom{n}{\lfloor n/2 \rfloor}$, with equality when $t_c = t = \lfloor n/2 \rfloor$. Thus, at $n = 128$, the largest bound occurs at $t_c = t = 64$. For the Curve25519 scalar field in (1), this gives

$$\Pr[\text{LDVR success}] \le Q\frac{\binom{128}{64}}{p} \approx Q \cdot 3.30954 \cdot 10^{-39} \approx Q \cdot 2^{-127.83}. \tag{7}$$

This bound applies to every admissible threshold and corruption budget at $n = 128$. Even with 2^{40} oracle queries, success is at most approximately $2^{-87.83}$.

Goldberg, Komlo, and Tessaro [5] discuss the corresponding bound for $p \approx 2^{256}$, identifying $n \le 131$ as a regime with success probability at most $Q/2^{128}$ for every threshold. Our field instead has $p \approx 2^{252}$, so we use the numerical bound in (7).

3 Attacks on Canonical Evaluation Points

3.1 Local subset sums and the Chinese remainder theorem

Fix a positive integer L and a collection $\mathcal{Q}$ of pairwise coprime moduli. The moduli need not be coprime to L. Set $P = \prod_{q\in\mathcal{Q}} q$ and $D = LP$, with $p \nmid D$. Choose local index sets $J_q \subseteq \{i \in [n] : i \mid Lq\}$ and a completion index set $R_0 \subseteq \{i \in [n] : i \mid L\}$, with all these sets pairwise disjoint.

For $i \in J_q$, define the integer weight $w_{q,i} = Lq/i$. Its scaled reciprocal is $D/i = (P/q)w_{q,i}$, which vanishes modulo every other modulus in $\mathcal{Q}$. This allows us to handle the congruence modulo each q through a separate

local subset sum problem. The construction includes the indices $i = qr$ with $r \mid L$ used in [6] and also permits prime-power moduli.

We build each local table by enumerating subsets of J_q and retaining representations whose integer sums lie in a chosen short interval. Let $\mathcal{A}_q \subseteq \mathbb{Z}/q\mathbb{Z}$ be the nonempty set of residues represented in the table. For each $a \in \mathcal{A}_q$, store a family $\mathcal{C}_q(a)$ containing one subset for each retained subset size. Every stored subset $C \in \mathcal{C}_q(a)$ satisfies

$$s_q(C) = \sum_{i \in C} w_{q,i}, \qquad s_q(C) \equiv a \pmod{q}.$$

Write $H_q(a) = \{|C| : C \in \mathcal{C}_q(a)\}$ for the available sizes. A residue outside $\mathcal{A}_q$ has no stored representation, although subsets realizing that residue may exist.

For $r \in R_0$, the scaled weight is $D/r = P(L/r)$. These indices therefore adjust the total sum without changing any local congruence. A dynamic program implemented with bitsets records, for each subset size k, all integer sums $\sum_{r \in B} L/r$ attainable by subsets $B \subseteq R_0$ with $|B| = k$. Suppose that every integer in a common interval $[u, v]$ is attainable for every completion size $k \in [k_-, k_+]$.

Define lower and upper bounds on the total scaled weight obtained from the local tables:

$$B_- = \sum_{q \in \mathcal{Q}} \frac{P}{q} \min_{\substack{a \in \mathcal{A}_q \\ C \in \mathcal{C}_q(a)}} s_q(C), \qquad B_+ = \sum_{q \in \mathcal{Q}} \frac{P}{q} \max_{\substack{a \in \mathcal{A}_q \\ C \in \mathcal{C}_q(a)}} s_q(C).$$

Choose parameters such that the target interval

$$\mathcal{I} = [B_+ + uP,\ B_- + vP] \subseteq [0, p-1] \tag{8}$$

is nonempty. For any target $A \in \mathcal{I}$, a total scaled weight B obtained from the local tables and satisfying $B \equiv A \pmod{P}$ leaves an integer quotient $(A - B)/P$ in $[u, v]$. The dynamic program can therefore supply the remaining contribution whenever the required completion size lies in $[k_-, k_+]$.

Lemma 2 (Combining the local subsets). *Let $A \in \mathcal{I}$ be an integer target, and define $a_q = A(P/q)^{-1} \bmod q$, where the inverse is taken modulo q. Suppose $a_q \in \mathcal{A}_q$ for every $q \in \mathcal{Q}$. If*

$$k = k_0 + \sum_{q \in \mathcal{Q}} h_q, \qquad k_0 \in [k_-, k_+], \qquad h_q \in H_q(a_q),$$

then the local tables and the completion dynamic program construct a subset C of exactly k signer indices satisfying $\sum_{i \in C} D/i = A$ over the integers.

Proof. For each q, select a stored subset $C_q \in \mathcal{C}_q(a_q)$ of size h_q. Their total scaled weight is $B = \sum_{q \in \mathcal{Q}} (P/q) \cdot s_q(C_q)$. Modulo each q, the corresponding local subset contributes A, while all other groups contribute zero. The Chinese remainder theorem therefore gives $B \equiv A \pmod{P}$.

Since $B_- \le B \le B_+$ and $A \in \mathcal{I}$, the bounds in (8) imply $u \le (A - B)/P \le v$. The dynamic program for R_0 therefore returns a subset C_0 of size k_0 satisfying

$$\sum_{r \in C_0} \frac{L}{r} = \frac{A - B}{P}.$$

Its scaled weight is $A - B$. Set $C = C_0 \cup \bigcup_{q \in \mathcal{Q}} C_q$. Disjointness of the index sets gives $|C| = k$, and their combined scaled weight is A. □

For a prescribed total size k, a second dynamic program finds compatible local sizes h_q and a completion size k_0 whenever such a choice exists. A sufficient condition for every integer size in

$$\left[k_- + \sum_q \min H_q(a_q), k_+ + \sum_q \max H_q(a_q) \right]$$

to be available is that consecutive sizes in each $H_q(a_q)$ differ by at most $k_+ - k_- + 1$. Indeed, adding a local size set to the completion interval produces shifted intervals that overlap or are adjacent. Their union

remains an integer interval and cannot become shorter, so the argument applies successively to the remaining local groups.

Retaining several local sizes may narrow the target interval in (8), because B_- and B_+ must cover all stored representations. We account for this tradeoff when selecting the parameters.

Now choose a set F of additional indices disjoint from R_0 and all local groups. For a partial subset $U \subseteq F$ and a challenge c, define the remaining target

$$A = \left(Dc - \sum_{i \in U} \frac{D}{i} \right) \bmod p, \tag{9}$$

where the expression is evaluated in $\mathbb{F}_p$ and A denotes its least nonnegative integer representative. If Lemma 2 applies with $k = t - |U|$, it constructs a completion C. The subset $S = U \cup C$ then has size t and satisfies $\sum_{i \in S} i^{-1} = c$ in $\mathbb{F}_p$, since $p \nmid D$. Hence, every accepted output solves the corresponding LDVR instance.

3.2 A generalized birthday attack over many challenges

Partition F into seven blocks $F_1, \ldots, F_7$. For each block, enumerate all scaled reciprocal sums $\sum_{i \in U_b} D/i$ with $U_b \subseteq F_b$, recording the selected indices. The lists include subsets of different sizes; the local reconstruction step enforces the required total size t.

Let $h_0 = \lfloor (\min \mathcal{I} + \max \mathcal{I})/2 \rfloor$. For Q independent uniform challenges, define

$$z_j = h_0 - Dc_j \pmod p, \qquad j \in [Q]. \tag{10}$$

An eighth list stores these transformed targets and their challenge indices. Since $p \nmid D$, the transformation is a bijection of $\mathbb{F}_p$, so the targets remain independent and uniform.

For selected subsets U_b, write $U = \bigcup_{b=1}^{7} U_b$, and let A be the remaining target defined in (9). Then

$$z_j + \sum_{b=1}^{7} \sum_{i \in U_b} \frac{D}{i} = h_0 - A \quad \text{in } \mathbb{F}_p.$$

Thus, finding a sum whose signed representative lies in

$$h_0 - \mathcal{I} = [h_0 - \max \mathcal{I},\ h_0 - \min \mathcal{I}]$$

is equivalent to finding a remaining target $A \in \mathcal{I}$.

We process the eight lists using a merge tree following Wagner's generalized birthday algorithm [7]. At each level, we sort by signed approximate values and retain pairwise sums in a prescribed interval. These intervals become progressively smaller, ending with $h_0 - \mathcal{I}$, slightly enlarged to account for rounding. The merged entries retain the selected indices and the associated challenge index.

For each surviving candidate, we recompute A in $\mathbb{F}_p$, check that its least nonnegative representative lies in $\mathcal{I}$, and apply Lemma 2 with $k = t - |U|$. If reconstruction succeeds, we verify that the completed subset has size t and satisfies the reciprocal subset sum equation.

The subset sum lists have substantial arithmetic structure, and intermediate filters may discard valid representations. The usual analysis based on independent random lists therefore does not directly apply, and we do not prove a success probability bound for this search. Section 3.5 states an explicit heuristic model for the search and compares its predictions with the measured counts.

Filtering residues before enumerating final pairs. To reduce the number of pairs considered at the final merge, we track residues alongside the approximate sums. Each entry occupies 32 bytes: a 64-bit signed approximate value, a 32-bit residue, padding, and a 128-bit record identifying the selected indices and the associated challenge.

The approximate value represents the signed field sum in units of 2^{188}. The residue records its signed integer representative modulo $M = 139 \cdot 137 \cdot 131 \cdot 127$. Whenever a merge adds or subtracts p to obtain the chosen representative, we apply the same adjustment to the residue modulo M.

Before the final merge, we group entries in the two remaining lists by their residues modulo 139. The corresponding local table contains only eight residues, so we enumerate only pairs compatible with this table. For surviving pairs, we check the moduli 137, 131, and 127, recompute the remaining target in $\mathbb{F}_p$, and apply the remaining local residue and subset-size checks. In the implementation, the final merge interval is enlarged by 16 units of 2^{188} at each endpoint. Every surviving candidate is then checked using its recomputed field value and the original reconstruction conditions.

3.3 Choosing the corruption budget

For tables storing one subset per residue, we choose the corruption budget by counting combinations of local residue choices and partial subsets $U \subseteq F$ that satisfy the final size requirement. If the selected local subsets have sizes h_q, the required completion size is $t - |U| - \sum_q h_q$, which must lie in $[k_-, k_+]$. We compute these counts by convolving the local subset-size distributions with $\binom{f}{j}$, where $f = |F|$ and $\binom{f}{j}$ counts the partial subsets of size j.

For the original 488-signer tables, the fraction satisfying this requirement is maximized at $t = 173$ and equals 0.782453. The stored local subsets favor smaller sizes, so $t = \lfloor n/2 \rfloor$ need not optimize the reconstruction procedure. These fractions quantify compatibility with the subset-size constraint alone. Under the model of Section 3.5, the fraction 0.782453 is used as the conditional probability that a candidate passing all residue checks has an admissible completion size.

Replacing stored subsets by their complements within the groups for moduli 41, 43, and 47 preserves the number of stored residues and the width of the target interval. We also include 59 fixed indices unused by the original construction and subtract their scaled contribution from the target. This gives size-244 solutions with a fraction of 0.782584, approximately the same as before. Starting from the six successful search candidates, we first reconstruct size-177 subsets for the original challenge, then complement these three local groups and add the fixed indices to obtain six size-244 subsets. We check that the local tables with the three complemented groups and the full completion dynamic program reconstruct the local and completion parts of each of these subsets. The completion parts have size 36 or 38, outside the common guaranteed range 18–35. For each resulting subset, we define a new challenge as its reciprocal sum in $\mathbb{F}_p$. These checks do not measure success on fresh challenges or establish a running time improvement.

For the multi-size tables, let M_t count pairs consisting of a partial subset $U \subseteq F$ and a local residue tuple for which Lemma 2 guarantees a completion of size $t - |U|$. Each residue tuple determines one congruence class for A modulo P. Writing $W = |\mathcal{I}|$, that class occurs either $\lfloor W/P \rfloor$ or $\lceil W/P \rceil$ times in $\mathcal{I}$. Hence the number N_t of guaranteed pairs (U, A) satisfies

$$\lfloor W/P \rfloor \, M_t \le N_t \le \lceil W/P \rceil \, M_t.$$

Under the condition on consecutive local sizes stated after Lemma 2, the attainable total sizes for each pair form an integer interval. We compute the distributions of its lower and upper endpoints using separate convolutions of the minimum and maximum local sizes with $\binom{f}{j}$. Then M_t equals the number of lower endpoints at most t minus the number of upper endpoints strictly below t.

For each local residue, we choose as center an available subset size closest to $\alpha|J_q|$, breaking ties in favor of the smaller size, for $\alpha \in \{0.4, 0.45, 0.5, 0.55, 0.6\}$. We also test the originally stored size as center. Around each center, we retain the available sizes within distance $d \in \{0, 1, 2, 3\}$. Across five configurations using $n \in \{450, 470, 488\}$ and $L \in \{2520, 5040, 27720\}$, optimizing t increases the lower bound $\lfloor W/P \rfloor \, M_t$ by approximately 21.8–23.7% relative to the original single-size tables and 10.3–14.6% relative to the best single-size tables in this sweep. Different pairs may correspond to the same challenge, so these gains do not establish a higher attack success probability or a faster search.

3.4 Concrete parameters and experiments

Table 1 gives the parameters of our 488-signer attack. We construct the local tables by exhaustive enumeration and use the dynamic program for R_0 to verify coverage of the required completion interval. Rebuilding and validating the tables takes 3.10 seconds. The implementation includes the tables, their construction algorithm, and the stored subsets.

Table 1. Parameters of the canonical attack over the scalar field in (1).

Parameter	Value
Signers / corruptions / signing threshold	488 / 173 / 174
Auxiliary integer L	27,720
Local moduli / completion indices	33 / 53
Completion subset sizes	$18, \ldots, 35$
Sizes of blocks $F_1, \ldots, F_7$	$5, 15, 15, 15, 15, 15, 14$
Challenge-list size	$Q = 2^{25}$
First / second merge reduction	5 / 24 bits
Final merge interval	$h_0 - \mathcal{I}$, enlarged for rounding

Table 2. Measured canonical subset search. Each batch contains 2^{25} targets. Timings exclude challenge hashing and signature assembly. The six $n = 488$ batches are the first six batches of the 64-batch experiment of Table 4, which gives predicted and measured counts.

Measurement	$n = 488$, $t = 173$	$n = 486$, $t = 172$
Successful batches / batches tested	1/6	0/6
Distinct successful challenges	1	0
Median batch time (seconds)	40.60	32.22
Total time for six batches (seconds)	250.59	194.16
Peak memory (GiB)	6.503	5.503

For the 488-signer experiment, we evaluate six preselected batches generated with seeds 400001 through 400006. Each batch contains pseudorandom targets, which we convert to simulated field challenges by inverting (10). No solutions are planted. Only the fourth batch succeeds, returning six size-173 subsets for one challenge. Their reciprocal sums are equal over the rationals, so we count one distinct successful challenge; Section 3.5 explains why accepted outputs arrive in such families. A separate verification script checks each subset and all $t + 1 = 174$ scalar equalities in the LDVR representation using the coefficients in (4).

The successful fourth batch takes 40.8908 seconds. Its final stage examines approximately 632 million compatible pairs, of which 141,586 survive the four-prime residue check and the target-interval test. Processing the first four batches takes 172.6552 seconds and uses 2^{27} simulated challenge targets. The full six-batch experiment uses $6 \cdot 2^{25} = 201{,}326{,}592$ targets.

The implementation runs on one CPU thread of an AMD EPYC 9V74 virtual machine and is compiled with GCC using `-O3`. Peak memory is measured as the process's maximum resident set size.

These experiments demonstrate a successful search at 488 signers. The reported timings reflect a batched search that shares computations across its 2^{25} challenges.

Sixty-four batches. To measure the behavior of the search beyond six batches, we run 64 batches with seeds 400001 through 400064 using a modified version of the implementation that records intermediate results. The first six use the same inputs as the batches of Table 2. This version records every candidate surviving the four-prime residue check and the target-interval test, stores the challenge targets in a memory-mapped file, and processes independent merge pairs in reverse order. The last two changes reduce peak memory usage without changing the arithmetic, the residue and size checks, or the accepted outputs. On these six inputs, it reproduces the compatible-pair counts, surviving-candidate counts, and accepted subsets of Table 2 exactly. Running the same 64 inputs on a second machine gives identical counts and outputs. Compiled with GCC 13.3 using `-O3` and run on one CPU thread of an AMD EPYC 9V74 virtual machine, this version takes a mean of 42.42 seconds per batch and a median of 41.75 seconds. The first batch takes 78.60 seconds, while the remaining batches take between 38.98 and 48.22 seconds. Peak resident memory is 5.004 GiB. These timings include writing the target file and the records of surviving candidates, and exclude challenge hashing.

The 64 batches enumerate 40,442,152,250 compatible pairs, of which 9,075,482 survive the four-prime residue check and the target-interval test. Of these, 28 candidates pass the residue checks of all 33 moduli, and

23 have an admissible completion size and are accepted. The accepted subsets solve six distinct challenges, in the batches with seeds 400004, 400036, 400047, 400049, 400051, and 400062, with 6, 1, 2, 4, 8, and 2 accepted subsets, respectively. Within each batch, the accepted subsets have equal reciprocal sums over the rationals. All 23 subsets pass the verification of the 174 scalar equalities described above. Section 3.5 compares these counts with predictions from an explicit heuristic model.

Hash cost. For the Ed25519 ciphersuite, FROST computes

$$c = \mathrm{LEInt}\big(\mathrm{SHA512}(\mathrm{enc}(R) \| \mathrm{enc}(\mathsf{PK}) \| m)\big) \bmod p$$

as specified in [2]. Distinct messages give distinct queries. Two single-threaded 2^{25}-query benchmarks on 96-byte inputs take 8.86–9.11 seconds for hashing and scalar reduction, or 19.20–20.38 seconds including target conversion (10) and writes to a circular memory buffer. Valid group encodings and distinct 32-byte messages are used; sampled outputs are independently checked.

Adding target generation to the successful canonical search gives an estimated 60.1–61.3 seconds for that batch, or 249–254 seconds across the first four batches. These estimates add separately measured times, and the cost of the search's own pseudorandom target generation is still included. No successful canonical search using actual hash challenges was timed. Interpolation, corruption access and signature assembly are excluded. The benchmarks use OpenSSL 3.0.13 and libsodium 1.0.18 on the AMD processor model above.

Attempts at smaller canonical configurations. For $n = 450$, $L = 2520$, and $t = 225$, the multi-size reconstruction algorithm succeeds on all 200 tested inputs sampled from the covered interval and residues represented in the local tables. We verify the size and reciprocal sum of every resulting subset. These tests assess reconstruction on covered targets; they do not measure search success on fresh challenges.

The local reconstruction procedures for the earlier 450- and 460-signer configurations each passed 1,000 checks on constructed covered targets, with the required subset size chosen from the admissible completion sizes. Separate 2^{25}-target searches with $(n, t) = (450, 166)$ and $(460, 168)$ found no solution. These searches took 68.12 and 77.94 seconds, respectively. The available historical hardware provenance and its limitations are documented in `HISTORICAL_ENVIRONMENTS.md` in the accompanying artifact. Among the configurations tested, 488 remains the smallest canonical signer count at which our search finds a successful subset.

3.5 A heuristic model for the search

The search of Section 3.4 returns only valid solutions, but we do not prove its success probability. We state an explicit heuristic model for the search, derive the counts it predicts at each stage, and compare them with the measured counts.

Hypothesis 3 (Residue and subset-size model). *For the compatible pairs at the final merge, the local residues a_q of the remaining target for $q \in \{137, 131, 127\}$ are independent and uniform.*

Conditional on surviving these checks and the target-interval test, the local residues a_q for the remaining 29 moduli are independent and uniform, and $|U|$ has distribution $\mathrm{Bin}(94, 1/2)$ independently of them. Conditional on passing all residue checks, the residues a_q, for all $q \in \mathcal{Q}$, are independent and uniform over their respective sets $\mathcal{A}_q$, and independent of $|U|$.

The same assumptions on the local residues for the remaining 29 moduli apply to the families of surviving candidates defined below, each family counted once.

We do not prove these assumptions. They concern the surviving candidates and their families, and they do not follow from the uniformity of the challenges. The lists have arithmetic structure, and the intermediate filters may discard valid representations, so the analysis of Wagner's algorithm for independent random lists [7] does not directly apply. The model is used only to derive heuristic predictions and compare them with the measured counts.

Table 3. Residues represented in the local tables of the 488-signer configuration, and the measured fraction of the 9,075,482 surviving candidates of the 64 batches whose local residue $a_q = A(P/q)^{-1} \bmod q$ is represented. The seventeen moduli $11, 13, 17, 19, 23, 25, 27, 29, 31, 32, 37, 41, 43, 47, 53, 59, 61$ represent every residue. For 127, 131, 137, and 139, every surviving candidate passes by construction.

q	$\lvert\mathcal{A}_q\rvert$	$\lvert\mathcal{A}_q\rvert/q$	measured	q	$\lvert\mathcal{A}_q\rvert$	$\lvert\mathcal{A}_q\rvert/q$	measured
67	64	0.9552	0.9554	101	16	0.1584	0.1585
71	52	0.7324	0.7324	103	16	0.1553	0.1555
73	43	0.5890	0.5893	107	16	0.1495	0.1495
79	52	0.6582	0.6581	109	16	0.1468	0.1469
83	29	0.3494	0.3490	113	16	0.1416	0.1416
89	32	0.3596	0.3595	127	8	0.0630	—
97	31	0.3196	0.3193	131	8	0.0611	—
				137	8	0.0584	—
				139	8	0.0576	—

Consequences. Under Hypothesis 3, a compatible pair passes the residue checks modulo 137, 131, and 127 with probability $(8/137)(8/131)(8/127) \approx 2.2463 \cdot 10^{-4}$. This estimate ignores the subsequent rejection of pairs whose recomputed sums lie outside the original final merge interval. The interval used during the search is slightly enlarged to account for rounding, increasing its width by approximately 0.02%.

Let

$$\pi = \prod_{q \in \mathcal{Q} \setminus \{127,131,137,139\}} \frac{|\mathcal{A}_q|}{q} \approx 8.32903 \cdot 10^{-7} \approx 2^{-20.20}.$$

Given N candidates surviving the four-prime residue check and the target-interval test, the predicted number passing all residue checks is $N\pi$. Under the conditional assumptions on the local residues and $|U|$, the fraction $a \approx 0.782453$ from Section 3.3 is the probability that a candidate passing all residue checks has an admissible completion size. The predicted number of accepted subsets is therefore $N\pi a$. These are expectations in the model; they do not assert independence between different candidates.

Families of related candidates. Surviving candidates are not independent. Several additional indices share a large common factor: 49, 98, 147, 196, 245, 392, and 441 are multiples of 49, and

$$\frac{1}{147} + \frac{1}{196} - \frac{1}{245} - \frac{1}{441} = \frac{1}{180}, \qquad \frac{1}{49} - \frac{1}{392} = \frac{1}{56},$$

where 180 and 56 divide L. Such exchanges change the remaining target by a completion weight, which is a multiple of P. We call two surviving candidates related if they have the same challenge index and their remaining targets are congruent modulo P. Related candidates form a family. They have identical local residues, so they pass or fail the residue checks together, but leave different completion targets and may require different completion sizes.

If there are F surviving families, the assumption on families in Hypothesis 3 predicts $F\pi$ families passing all residue checks. Linearity of expectation does not require independence between families. A passing family need not contain an accepted subset, and different families can correspond to the same challenge. For the six successful challenges in this experiment, the accepted subsets for each challenge belong to one family.

Comparison with the measurements. Table 4 compares the model predictions with the measured counts. The total number of surviving candidates differs from the prediction by approximately 0.1%, and the relative difference is at most 1.21% in every batch. The measured pass rates of the individual residue checks in Table 3 differ from the fractions $|\mathcal{A}_q|/q$ by at most 0.121% in relative terms. For example, 98,866 candidates pass the seven checks modulo 67 through 97, compared with approximately 98,835 predicted by the product of the fractions $|\mathcal{A}_q|/q$. The distribution of $|U|$ has total variation distance 0.0007 from $\mathrm{Bin}(94, 1/2)$.

Seven families pass all residue checks, compared with a predicted mean of 2.97. They contain 28 candidates, of which 23 yield accepted subsets for six distinct challenges. At seed 400059, both candidates require 36 completion indices, outside the permitted range 18–35. Passing families contain more candidates than the average surviving family, whose size is approximately 2.5.

Table 4. Heuristic predictions and measured counts for the 64 batches of 2^{25} challenges. Predictions use Hypothesis 3 and the measured numbers of compatible pairs and families. The prediction for the surviving candidates neglects the enlargement of the final merge interval.

Stage	Predicted	Measured
Compatible pairs at the final merge	—	40,442,152,250
Surviving candidates	9,084,659	9,075,482
Surviving candidates, per batch	141,774–142,087	140,196–143,706
Families among the surviving candidates	—	3,566,135
Mean of $\lvert U\rvert$ over surviving candidates	47.000	47.001
Standard deviation of $\lvert U\rvert$	4.848	4.848
Families passing all residue checks	2.97	7
Candidates passing all residue checks	7.56	28
Accepted subsets	5.91	23
Distinct successful challenges	—	6

If we additionally model the family pass events as independent Bernoulli variables with common probability π, their count is binomial with parameters F and π and is well approximated by a Poisson distribution with mean 2.97. This additional model gives probability approximately 3.2% for seven or more passing families. Independence between families is unproved and does not follow from the assumptions in Hypothesis 3; this tail calculation is only a heuristic comparison. The measurements agree closely with the model for the early filters; at the final stage, the accepted subsets arrive in families, as explained above, and in larger numbers than predicted. The measurements do not establish the model or a success probability for a fresh batch. The model predicts approximately 0.046 families passing all residue checks per batch. We give no prediction for the number of distinct successful challenges in Table 4, since this also depends on completion and on whether different passing families share a challenge.

Proved baseline and counting bound. We distinguish the model predictions from quantities obtained by counting. Without the additional indices F, the deterministic construction tests whether $Dc \bmod p$ lies in the set of targets guaranteed by Lemma 2. Let $M_t^{(0)}$ be the number of local residue tuples for which the lemma guarantees a completion of size t, that is, M_t of Section 3.3 restricted to $U = \emptyset$. Lemma 2 gives a per-challenge success probability of at least

$$\frac{\lfloor W/P \rfloor \, M_t^{(0)}}{p}, \qquad W = |\mathcal{I}|.$$

For the tables of Table 1 and $t = 173$, this lower bound is approximately $5.37105 \cdot 10^{-38}$, or $2^{-123.81}$. It concerns guaranteed targets in the chosen interval and does not count every target representable by these indices. By contrast, the 5,020-signer instantiation of [6] has a proved per-challenge success probability of approximately 1/194. Our birthday search demonstrates success with 488 signers, but we do not prove its success probability.

We adapt the reciprocal-query counting argument of [6, Theorem 5.2], stated there for $t = n/2$. For general t, rational subset sums lie in $[H_n - H_{n-t}, H_t]$, where $H_k = \sum_{i=1}^{k} 1/i$. Using unconstrained local residue sets as supersets gives an upper bound $\Delta_t(n)$ on the number of challenges c for which the reciprocal subset sum equation has a solution with $|S| = t$. Thus, any reciprocal-query attack with Q uniform challenges satisfies

$$\Pr[\text{success}] \le \min\{1, Q\Delta_t(n)/p\}.$$

At $n = 488$ and $t = 173$, $\log_2 \Delta_t(n) \approx 259$ exceeds $\log_2 p$, so this bound is not informative over our field. A value below 1 limits the success probability and does not exclude success. For fixed n, the residue factor in this bound does not depend on t, and the width of the interval of rational subset sums is symmetric in t and $n-t$ and largest at $t \in \{\lfloor n/2 \rfloor, \lceil n/2 \rceil\}$. Hence the value of $\Delta_t(n)$ at $t = \lfloor n/2 \rfloor$ bounds the success probability for every corruption budget at that signer count. The bound is approximately 0.303 for $(n, t, Q) = (468, 234, 1)$ and 0.939 for $(n, t, Q) = (418, 209, 2^{25})$. These values do not establish a sharp minimum signer count for this attack strategy.

Remark 4 (Asymptotics). Our results improve the concrete parameters over a fixed field. We do not claim an asymptotic improvement over the $n = \Theta(\log^2 p)$ signers of [6], and we do not prove a success probability for the birthday search. Numerical evaluations of the counting bound at finitely many field sizes do not by themselves establish an asymptotic lower bound on the number of signers.

4 Adversarially Chosen Evaluation Points

The evaluation points are chosen adversarially and fixed before any oracle challenges are sampled. We give their reciprocals a binary structure and add m additional values $a_1, \ldots, a_m$. For $n = B + 1 + m$, define

$$\begin{aligned}(v_1, \ldots, v_n) &= (0, 2^0, 2^1, \ldots, 2^{B-1}, a_1, \ldots, a_m),\\ w_i &= 1 + v_i, \qquad x_i = w_i^{-1}.\end{aligned} \tag{11}$$

All arithmetic defining the points is performed in $\mathbb{F}_p$. We require the reciprocals w_i to be distinct and nonzero, ensuring that the x_i are valid evaluation points.

The shift by 1 maps the zero-valued index to the nonzero reciprocal $w_1 = 1$ and adds exactly t to every size-t subset sum. Consequently, a subset S of size t satisfies (3) if and only if $\sum_{i \in S} v_i = c - t$ in $\mathbb{F}_p$.

4.1 A query-optimal binary family

Assume $2^B < p$. For each $J \subseteq [m]$, compute the least nonnegative representative $y = (c - t - \sum_{j \in J} a_j) \bmod p$. If $y \geq 2^B$, discard this choice. Otherwise, compute $r = \mathrm{wt}(y)$, the number of 1 bits in its binary expansion, and accept if

$$r = t - |J| \quad \text{or} \quad r = t - |J| - 1. \tag{12}$$

The resulting subset contains the additional indices corresponding to J and the power-of-two indices identified by the binary expansion of y. Include the index corresponding to $v_1 = 0$ precisely in the second case. In either case, the subset has exactly t indices and satisfies (3).

Conversely, every size-t solution determines a choice of J and a binary contribution in $[0, 2^B - 1]$. Since $2^B < p$, this contribution equals the computed representative y, and its Hamming weight satisfies (12). Thus, the algorithm recognizes every successful challenge and recovers a valid subset whenever one exists. It tests at most 2^m choices per challenge.

Proposition 5 (Optimal binary family). *Let $n = B + 1$, $2^B < p$, and $1 \leq t < n$. With no additional values, the points in* (11) *admit an efficient subset search algorithm with one-query success probability $\rho = \binom{n}{t}/p$. For Q independent uniform challenges, the attack succeeds with probability $1 - (1 - \rho)^Q$. Hence it attains the unconditional LDVR upper bound in* (6) *for every query budget when $t_c = t$.*

Proof. Every sum of selected v_i lies in $[0, 2^B - 1]$, so distinct integer sums remain distinct modulo p. The binary expansion determines the selected powers of two, and the prescribed subset size determines whether the zero-valued index is included. Consequently, distinct size-t subsets yield distinct sums and therefore distinct challenges after adding t modulo p.

There are exactly $\binom{n}{t}$ such subsets, and the algorithm accepts precisely their corresponding challenges. This gives the stated one-query probability. Independence of the challenges gives the Q-query probability. □

For $B = 252$, $n = 253$, and $t = 126$, the one-query success probability is approximately 0.100028957989. The common shift by 1 allows the zero-valued index to correspond to the valid evaluation point $x_1 = 1$.

4.2 Query bounds and SHA-512 experiments

For $2 \leq n \leq 253$, choose $B = n - 1$, $m = 0$, and $t = \lfloor n/2 \rfloor$. Proposition 5 gives the exact success probability after Q queries in the uniform-oracle model and the minimum query budget Q_{99} required for 99% success:

$$P_Q = 1 - (1 - \rho_n)^Q, \qquad \rho_n = \frac{1}{p} \cdot \binom{n}{\lfloor n/2 \rfloor}, \qquad Q_{99} = \left\lceil \frac{\log(0.01)}{\log(1 - \rho_n)} \right\rceil. \tag{13}$$

Table 5. Measured SHA-512 challenge searches. Timings include hashing, scalar reduction, and the acceptance test.

n	t	Queries	Seconds	Outcome
230	115	16,844,106	7.3059	Subset found
228	114	300,000,000	142.4899	No subset found
228	114	60,128,928	26.1114	Subset found

The attack attains the upper bound in (6). Since the central binomial coefficient maximizes $\binom{n}{t}$, this choice also maximizes the attainable success probability over all degrees with $t_c = t$. Thus, for fixed n and Q, no choice of evaluation points or adaptive query coefficients achieves a larger success probability. This optimality concerns the number of oracle queries, including any made during preprocessing.

SHA-512 challenge search. We fix $R = g^7$ and $\mathsf{pk} = g^{11}$ and compute the Ed25519 challenge hash described in Section 3.4. Each message consists of a fixed 16-byte prefix, an 8-byte run identifier, and an 8-byte counter. Together with the two 32-byte group encodings, this gives a 96-byte hash input. Distinct run-identifier and counter pairs give distinct inputs.

The timed loop computes SHA-512, interprets its output as a little-endian integer, and reduces it modulo p to obtain c. It then computes $y = (c - t) \bmod p$ and tests whether $y < 2^{n-1}$ and $\mathrm{wt}(y) \in \{t, t-1\}$.

Table 5 reports all three runs. At $n = 228$, the first run finds no solution within 300 million queries, while the second finds a successful subset after 60,128,928 queries. The two runs use distinct inputs and require 360,128,928 queries and 168.6013 seconds in total. The 230-signer run finds a successful subset in 7.3059 seconds.

The experiments run on one CPU thread of an Intel Xeon Platinum 8573C. The C++ implementation is compiled with GCC 13.3 using `-O3` and links against OpenSSL 3.0.13 and libsodium 1.0.18. The canonical experiments use different hardware, so their running times are not directly comparable.

Verification and other costs. A separate Python verification script checks the hash, evaluation points, selected subset, and all 115 scalar equalities in the LDVR representation at $n = 228$, using the coefficients in (4). We also generate a degree-114 Shamir instance with $a_0 = 11$, $a_1 = 7$, and deterministic remaining coefficients. Public interpolation from the share commitments recovers R, and the selected 114 shares produce the response z. The verification script also checks the Schnorr equation $g^z = R \cdot \mathsf{pk}^c$ in the Ed25519 prime-order subgroup. The successful 230-signer subset passes the corresponding 116-coordinate LDVR check.

The search timer excludes outputting the selected indices, point and key setup, computation of the coefficients λ_i, interpolation, signature assembly, and verification. Separately, generating public share commitments and interpolating R takes 0.037 seconds at $n = 228$; response assembly and group verification take less than 0.001 seconds. The experiment models corruption access by supplying shares locally. Participant access and communication are outside the measurements. The implementation includes the successful challenges, selected subsets, and verification code.

Estimated running times. At $n = 228$, the one-query success probability is $\rho_n \approx 3.14612912719 \cdot 10^{-9}$, giving $Q_{99} = 1{,}463{,}757{,}524$. A budget of 300 million queries gives approximately 61.09% success in the uniform oracle model. We do not infer a success probability from the two measured runs.

Table 6 gives the query budgets obtained from (13). For a measured throughput of r queries per second, we estimate the time to process the 99% query budget as Q_{99}/r. The reported ranges use the slowest and fastest measured rates, approximately 2.105 and 2.306 million queries per second. These estimates assume that the measured throughput can be sustained for the entire search; the longer searches have not been executed.

Under these throughput assumptions, a 14-day budget permits 99% success at $n = 218$ at either measured rate. The bound in (6) rules out that probability at $n = 217$ within either query budget. For a 30-day budget, the minimum signer count is 216 at the faster rate and 217 at the slower rate.

The probability guarantees above concern independent uniform field challenges. A fixed SHA-512 function is deterministic, and the experiments demonstrate successful inputs without establishing a probability

Table 6. Query budgets for 99% success in the uniform oracle model. Estimated times are obtained by dividing Q_{99} by the measured single-thread query rates.

n	t	Q_{99}	Estimated time
230	115	367,537,368	2.66–2.91 minutes
228	114	1,463,757,524	10.58–11.59 minutes
224	112	23,214,226,240	2.80–3.06 hours
220	110	368,103,832,653	1.85–2.02 days
218	109	1,465,722,533,660	7.36–8.06 days
216	108	5,835,996,143,201	29.30–32.08 days

Table 7. The three additional values defining the 256-point construction. Concatenate the two hexadecimal halves in each row to obtain a_j.

j	Most significant 128 bits	Least significant 128 bits
1	`018f36539832f692835835715f974c04`	`5edadc4ca37d75861016112312dffcc9`
2	`0b139e78c69411510871cd733875ef87`	`95cc4c33c1a0dbfc14c848643c142ce6`
3	`0ae9956194fcf7aa3d2b1c22995553fd`	`9bbd0ca5ee2ab041ef0965b8970bc268`

theorem for that function. In the random function model, distinct inputs give independent uniform 512-bit strings. Their reductions modulo p remain independent, with statistical distance from uniform at most

$$\frac{p}{4 \cdot 2^{512}} < 1.35 \cdot 10^{-79}.$$

4.3 Constant success with 256 evaluation points

Set $B = 252$, $m = 3$, and $t_c = t = 128$, giving $n = 256$. We fix the three additional values as in Table 7. The subset search algorithm tests all eight choices of additional indices for each challenge.

Theorem 6 (Explicit chosen-point attack). *Over the scalar field in* (1)*, fix the* 256 *evaluation points in* (11) *using Table 7. For a uniform challenge, the subset search algorithm finds a size-*128 *subset solving LDVR with probability at least* 0.51929337649*. With seven independent uniform challenges, its success probability is at least* 0.99406855*.*

Proof. The 256 reciprocals are distinct and nonzero, so they define valid evaluation points. For each $J \subseteq [3]$, set $s_J = \sum_{j \in J} a_j \bmod p$, $u_J = 128 - |J|$, and define

$$A_J = \left\{ y \in [0, 2^{252} - 1] : \mathrm{wt}(y) \in \{u_J, u_J - 1\} \right\}.$$

The challenges accepted for this choice of J form the translated set $T_J = 128 + s_J + A_J$ modulo p. The algorithm succeeds precisely on $\bigcup_{J \subseteq [3]} T_J$. Since $2^{252} < p$, translation preserves the cardinality of each set, giving

$$|T_J| = \binom{252}{u_J} + \binom{252}{u_J - 1} = \binom{253}{u_J}.$$

Consequently, Vandermonde's identity yields

$$\sum_{J \subseteq [3]} |T_J| = \sum_{\ell=0}^{3} \binom{3}{\ell} \binom{253}{128 - \ell} = \binom{256}{128}.$$

Let $I_{JK} = |T_J \cap T_K|$. The second-order Bonferroni inequality gives

$$\Pr[\text{success}] \geq \frac{\binom{256}{128} - \sum_{J<K} I_{JK}}{p}, \tag{14}$$

where the sum includes each unordered pair once. We compute all 28 intersections exactly using the dynamic program described below. For the values in Table 7,

$$\frac{1}{p}\sum_{J<K} I_{JK} = 0.2778123824851387\ldots,$$

which gives a lower bound of $0.5192933764931037\ldots$. Seven independent uniform challenges therefore give success probability at least

$$1-(1-0.51929337649)^7 > 0.99406855.$$

□

Computing the intersections. For a pair J, K, let δ be the least nonnegative representative of $(s_J - s_K) \bmod p$. An intersection corresponds to $y \in A_J$ and $z \in A_K$ satisfying either $z = y + \delta$ or $z = y + \delta - p$. These two integer cases are disjoint.

We count the first case by adding the nonnegative shift $d = \delta$ to y. For the second case, we exchange the roles of y and z and add $d = p - \delta$ to z, exchanging the permitted Hamming weights as well. A shift $d \geq 2^B$ contributes no pairs.

For each remaining shift, a dynamic program processes the bits from least to most significant. Its state records a carry $\kappa \in \{0, 1\}$, initially zero, and both accumulated Hamming weights. Let d_b be the current bit of d. For each input bit $\epsilon \in \{0, 1\}$, compute

$$\eta = (\epsilon + d_b + \kappa) \bmod 2, \qquad \kappa' = \left\lfloor \frac{\epsilon + d_b + \kappa}{2} \right\rfloor.$$

Here, η is the output bit and κ' is the carry passed to the next bit position. We increment the two weights by ϵ and η, respectively. After B bits, we retain states with carry zero and the permitted weights.

With maximum permitted weights u, v, the dynamic program has $O(B(u+1)(v+1))$ states. We validate it against exhaustive enumeration on three small fields. The intersection counts and code are included with the implementation.

A separate million-challenge experiment finds successful subsets for 568,250 challenges, an observed fraction of 56.825%. This measurement is separate from the proved lower bound. The attack tests at most eight candidates per challenge; the intersection computations are used to prove the bound and are not required to run the attack.

The binary construction in [6] uses approximately twice the modulus bit length in evaluation points for its probability-one attack. Our explicit construction gives constant success per challenge with roughly one evaluation point per bit.

Use of AI tools

The author proposed combining Chinese remainder theorem reconstruction with generalized birthday search over many challenges, using different list sizes in that search, and retaining multiple subset sizes in the local reconstruction tables. The author also developed the sharper finite-query upper bound for LDVR and the query-optimal binary family for $2 \leq n \leq 253$ when $t_c = t$, together with their proofs. The heuristic model of Section 3.5, its predictions, and their comparison with the measurements were developed by the author.

ChatGPT developed the constant-success construction with 256 evaluation points and the proof of its stated success bound. ChatGPT also proposed the corruption-budget optimization for the canonical-point attack. Under the author's direction and with the author's input, AI tools explored and selected concrete parameters, including the subset sizes retained in the local tables. The author independently checked the 256-point construction and its proof. The author also checked the corruption-budget analysis and the validity of the selected parameters.

AI tools further assisted with the implementation and execution of the experiments, running-time estimates, the preparation of Figure 1, and language editing. For the 64-batch experiment, AI tools assisted with the modified implementation that records intermediate results, the scripts computing the measured values in Tables 3 and 4, numerical checks, and the verification of the experimental records. ChatGPT 6 was used

primarily in the Astra Ultra configuration, with Sol Extra High additionally used for language editing. The experimental code and reported results were additionally reviewed with assistance from Claude in the Fable 5.1 Max configuration.

All other new ideas and proofs of new results in this paper were developed by the author. The author takes responsibility for all claims in the paper.

References

1. Bacho, R.: Experimental artifacts for Concrete LDVR Attacks with Fewer Signers. Zenodo (2026). https://doi.org/10.5281/zenodo.23474610, version 1.0.0
2. Connolly, D., Komlo, C., Goldberg, I., Wood, C.A.: The flexible round-optimized Schnorr threshold (FROST) protocol for two-round Schnorr signatures. RFC 9591 (2024), https://www.rfc-editor.org/rfc/rfc9591
3. Crites, E., Katz, J., Komlo, C., Tessaro, S., Zhu, C.: On the adaptive security of FROST. In: Kalai, Y.T., Kamara, S.F. (eds.) Advances in Cryptology – CRYPTO 2025, Part VI. Lecture Notes in Computer Science, vol. 16005, pp. 480–511. Springer (2025). https://doi.org/10.1007/978-3-032-01887-8_16
4. Galambos, J.: Bonferroni inequalities. The Annals of Probability **5**(4), 577–581 (1977). https://doi.org/10.1214/aop/1176995765
5. Goldberg, I., Komlo, C., Tessaro, S.: On FROST and unconditional LDVR security. Cryptology ePrint Archive, Paper 2026/2203 (2026), https://eprint.iacr.org/2026/2203
6. Rodríguez García, À., Tessaro, S.: Cryptanalysis of LDVR: Polynomial-time adaptive attacks against threshold Schnorr signatures. Cryptology ePrint Archive, Paper 2026/2126 (2026), https://eprint.iacr.org/2026/2126
7. Wagner, D.: A generalized birthday problem. In: Advances in Cryptology – CRYPTO 2002. Lecture Notes in Computer Science, vol. 2442, pp. 288–303. Springer (2002). https://doi.org/10.1007/3-540-45708-9_19