The Geometry of Zero Knowledge: Tight Masking and Efficient Multilinear Arguments

Xiang Wang[1] and Ji Xiang[2]

[1] Guizhou University, China
ee.wangx24@gzu.edu.cn
[2] Chongqing University, China
20240674@stu.cqu.edu.cn

17 November 2026

Abstract. Zero-knowledge proofs have become a fundamental tool for verifiable computation and privacy-preserving applications, driving growing interest in proof systems that are both succinct and efficient. In polynomial-based arguments, however, achieving zero knowledge introduces additional costs. Random masks must hide the evaluations and intermediate messages revealed during verification while preserving the underlying computation. Existing approaches range from protecting individual messages with cryptographic commitments to constructing small masking polynomials tailored to particular protocols. These techniques raise a more general question: how much randomness is needed when the verifier observes only a structured collection of evaluations and linear combinations?

We present GoZK, a query-aware approach to zero knowledge that constructs masking spaces from the verifier's conditional joint observations. We characterize these observations through affine images and develop low-degree masks for selected Cartesian strata, witness polynomials, and product-tree relations. In particular, two product trees that reveal eight boundary evaluations and a single root difference admit a nine-dimensional masking space, matching the dimension of their specified uniform observation interface. We integrate these constructions with a causal masking polynomial, a global sumcheck, and a pairing-based polynomial commitment scheme. The resulting protocol admits exact online simulation under the stated setup and private-coin conditions, with classical knowledge soundness established in a registered algebraic-group and random-oracle model. In a Rust implementation over BLS12-381, query-aware masking reduces the number of source-mask coefficients from 390 to 19 relative to an implemented projected arbitrary-point construction, saves 4,128 proof bytes, and improves prover time by factors of 1.40–1.67 on matched eight-column instances. Global sumcheck further reduces communication by 6,752–10,400 bytes, with a modest prover-time tradeoff.

Keywords: Zero knowledge · Query geometry · Sumcheck · HyperPlonk · Polynomial commitments

Contents

1 Introduction . . . 3
1.1 Technical Overview . . . 5
1.2 Contributions and Results . . . 6
1.3 Relation to Previous Work . . . 7
2 Relations, Observations, and Cost Parameters . . . 10
3 Geometry of Masking . . . 12
3.1 A running example with two product trees . . . 12
3.2 Conditional observations and interpolation . . . 12
3.3 A nine-coefficient mask for paired trees . . . 13
3.4 Witness masks and related source spaces . . . 15
4 The GoZK Protocol . . . 15
4.1 Native relations and terminal observations . . . 16
4.2 Causal masking of one global sumcheck . . . 17
4.3 Commitments and authentication of the observations . . . 19
4.4 The complete core protocol . . . 20
4.5 Scalar lookup extension . . . 20
5 Security . . . 20
5.1 Completeness and simulation setup . . . 20
5.2 Exact causal privacy . . . 21
5.3 Classical knowledge soundness . . . 23
6 Implementation and Concrete Costs . . . 24
6.1 Native tables and reusable commitments . . . 25
6.2 Communication from the observation interface . . . 25
6.3 Fewer relations can require more bytes . . . 26
7 Evaluation . . . 26
7.1 Experimental design . . . 26
7.2 Effects of the masking and sumcheck mechanisms . . . 28
7.3 End-to-end scale . . . 29
7.4 Engineering tradeoffs . . . 30
8 Conclusion . . . 30
References . . . 31
A Proofs for Query-Aware Masking . . . 36
B Security Proofs . . . 43
C Full Protocol Specification and Implementation Details . . . 52
D Additional Experiments and Reproducibility . . . 61

1 Introduction

Zero-knowledge proofs (ZKPs) [43] allow a prover to establish the validity of a computation without revealing the private information used to perform it. Once primarily a subject of theoretical cryptography, they have become an important building block for privacy-preserving transactions, verifiable computation, and succinct arguments. This transition has driven substantial progress in proof systems with short proofs and efficient verification. Yet the cost of generating a zero-knowledge proof remains a central challenge, particularly for large computations. Beyond proving that a relation holds, the prover must ensure that the messages exchanged during verification reveal nothing about the underlying witness.

Multilinear proof systems offer a natural setting for studying this cost. Systems based on multilinear extensions and the sumcheck protocol [62] represent computations as low-degree polynomial relations over the Boolean hypercube. The verifier checks these relations through a sequence of algebraic reductions, eventually obtaining a small number of polynomial evaluations at random points. Although this structure supports efficient verification, the intermediate sumcheck messages and terminal evaluations may depend on the private witness. A common approach to obtaining zero knowledge is to replace a polynomial P by $P + M$, where M is a random polynomial that vanishes on the Boolean hypercube. The masked polynomial preserves the original computation at every Boolean point while randomizing the values exposed elsewhere.

The choice of M, however, affects more than privacy. Its coefficients contribute to prover computation, its degrees influence the size of sumcheck messages, and its algebraic support determines the commitment parameters needed to authenticate evaluations. A mask that is sufficiently random for arbitrary queries may therefore be unnecessarily expensive for a protocol that reveals only a small, structured collection of values. At the same time, reducing the number of random coefficients without accounting for correlations between observations can leave information about the witness exposed.

Several approaches have addressed these difficulties. Early zero-knowledge transformations for polynomial protocols use random masking polynomials to hide the values returned by polynomial oracles [14,27]. Commitment-based arguments can instead check relations among hidden values. Bulletproofs, for example, verifies an inner-product relation on committed vectors [22]. Such techniques provide general mechanisms for hiding information, but their algebraic or cryptographic overhead can be substantial. Libra [88] made an important advance by showing that sumcheck messages can be protected with small random polynomials. Its construction exploits the limited number of messages actually revealed by the protocol. It also develops small masks for the evaluations exposed between consecutive layers of the GKR protocol [42]. HyperPlonk [26] extends this direction to multilinear polynomial IOPs. Its zero-knowledge compiler uses structured vanishing masks and a query restriction that ensures distinct evaluations can be independently randomized.

These results establish that the structure of the verifier's queries matters. They also suggest a further distinction. In a complete multilinear argument, the verifier does not necessarily observe a collection of independent evaluations of each committed polynomial.

Some evaluations lie on fixed Boolean slices, others share challenge coordinates, and some are disclosed only through linear combinations of several sources. Previously revealed messages can constrain the randomness available to later observations. The information that must be hidden is therefore determined by the joint observation interface, including the order in which its values become public.

A simple example arises in the product-tree checks used for permutation arguments. Consider two polynomial-encoded product trees A and B. At the end of a sumcheck, the verifier requires four non-root evaluations from each tree to check its leaf and internal relations. It must also verify equality of their products. A direct construction could independently mask five evaluations of each tree, using ten random coefficients. The protocol, however, does not require the two individual values at the terminal point. It needs only their difference, $(A - B)(r)$, at a common complete query point r. The resulting joint interface contains eight non-root evaluations and one difference, in total.

This distinction has an algebraic consequence. The nine observations can be made jointly uniform using a nine-dimensional space of low-degree masks. Eight coefficients independently cover the boundary observations, while one additional coefficient supplies the remaining randomness for the root difference. The dimension is tight for this specified full uniform interface. More importantly, the construction preserves the distinction between observing a difference and observing its two operands: the commitment layer authenticates $(A - B)(r)$ without disclosing $A(r)$ and $B(r)$ separately.

The example illustrates the central perspective of this paper. The masking requirement is determined by the information revealed to the verifier. The relevant structure is more precise than the number of evaluation requests. It includes the locations of those requests, linear dependencies between their answers, and the constraints imposed by earlier messages. Two observation maps may have the same rank but different affine supports, and hence different conditional distributions. Establishing zero knowledge requires the supports, including their translations, to agree across compatible private states.

Turning this perspective into a proof system raises two additional challenges. First, source evaluations are only part of the transcript. Sumcheck messages expose witness-dependent weighted sums, and their randomness must remain available as verifier challenges are generated. Second, the polynomial commitment scheme must preserve the intended observation interface. Repeated evaluations must remain consistent, derived linear combinations must be authenticated without revealing additional values, and commitment openings must not reintroduce dependencies on the witness. These requirements connect the local algebra of masking to the complete execution of the protocol.

We address these challenges in GoZK, a query-aware multilinear argument that constructs its masks from prescribed conditional observations. Our approach combines explicit low-degree interpolation spaces, a causal masking polynomial for sumcheck messages, and a commitment interface that preserves joint linear observations. We instantiate the construction for gate, copy, and public-input relations, with an extension to single-table scalar lookups. The resulting design links the randomness needed for

privacy to the polynomial degrees and authenticated evaluations that determine concrete proving costs.

1.1 Technical Overview

We begin with the observation space of a masked polynomial. A verifier's query does not have to range over every point of the ambient field. In multilinear arguments, a query often fixes some coordinates to Boolean values while leaving others at non-Boolean challenges. We describe these regions as Cartesian strata. For the Boolean domain, a coordinate belongs to one of three states: 0, 1, or a non-Boolean value. Given a prescribed collection of such strata, we construct a polynomial basis whose elements vanish on the Boolean cube and whose evaluations form an invertible matrix at one selected point from each stratum.

The construction uses the simple vanishing factor $\phi(t) = t(t-1)$. By combining these factors with Boolean interpolation polynomials, we obtain masks of individual degree at most two. Their dimension matches the number of independent uniform observations in the specified interface. Unlike a construction that relies on distinct values in one common coordinate, the interpolation uses the pattern of fixed and non-Boolean coordinates across the selected regions. The representative points need not share their non-Boolean coordinates, and different strata may expose different parts of the underlying polynomial.

For the paired-tree example, four basis directions cover the boundary evaluations of each tree. A fifth direction, $\phi(u)\phi(v)$, vanishes at every boundary query but remains nonzero at the terminal point of the root-equality check. Adding this direction to only one tree produces the required nine-dimensional joint mask. The same interpolation principle yields compact masks for witness and lookup sources. In a core argument with k witness columns, the complete witness-query interface uses $k + 2$ mask coefficients. With eight columns and the paired-tree construction, the core sources therefore require 19 coefficients in total.

The preceding dimension statements concern fixed observation interfaces. To analyze a sequence of adaptive observations, we consider the mask distribution conditioned on the transcript already revealed. Earlier linear observations restrict the originally uniform mask to an affine fiber. A subsequent response is uniformly distributed over the image of that fiber under its observation map. The new randomness available at that step is measured by the rank increase after incorporating the previous constraints. Privacy requires more than the correct rank: the affine image itself must be independent of the compatible private state. This characterization also gives an online sampling procedure when a representative and basis of the common image can be computed efficiently. Section 3 develops the general statement and its explicit interpolation instances.

We next integrate the source masks into the multilinear argument. The core protocol checks the gate constraints, two sets of product-tree leaves, two internal tree relations, equality of their products, and the public-input relation. These seven residuals are embedded into a common variable space and combined into a single sumcheck. Random linear combinations of sumchecks are already available in HyperPlonk [26]; here their use is coupled to the prescribed source observations and a common degree certificate.

The sumcheck itself introduces another form of information disclosure. Each round polynomial is a weighted sum of the underlying masked relations, and its coefficients depend on the witness. Following the small-mask approach developed in Libra [88], we use a helper polynomial formed from independent univariate components. Its coefficients are arranged so that each round receives fresh randomness unavailable to the earlier round messages. We additionally constrain the helper's two first-coordinate half-cube sums to zero. For honest per-axis degree caps $E_1, \ldots, E_L$, this yields $\sum_{a=1}^{L} E_a - 1$ transmitted round coefficients. The helper's final evaluation is recovered from the terminal sumcheck equation and authenticated through a commitment opening.

The commitment layer completes the construction. We use separate source and helper groups in a pairing-based polynomial commitment scheme. Its evaluation check builds on the quotient identity of Kate, Zaverucha and Goldberg [53]. Source commitments are independently blinded, and their openings are randomized with fresh coefficients. Linearity allows the verifier to authenticate a root difference through the derived commitment $C_A - C_B$. Evaluations are batched only when they share the same complete point and source group. The protocol fixes every observation before generating opening-batching challenges, preserving the dependencies required by the simulator.

These components give an exact online simulation of the complete transcript under the stated simulation setup, observation interface, and private-coin distribution. The simulator samples commitments, round coefficients, source observations, and openings from their respective conditional distributions. It uses the original challenge function without reprogramming it. Knowledge soundness is established separately for classical algebraic adversaries under the registered online algebraic-group and random-oracle model. The analysis uses the actual published commitment spaces, in the soundness calculation.

The construction also permits an explicit accounting of communication. Proof length depends on the sumcheck degrees, the number of transmitted terminal values, and the number of distinct opening buckets. These quantities need not decrease together. For example, eliminating the explicit public-input relation reduces the number of checks but may raise the degrees of the remaining expressions. Consequently, the smaller relation set can produce a larger proof. Section 6 derives these costs from the protocol, and Section 7 measures the contributions of source masking and sumcheck organization separately.

1.2 Contributions and Results

Query-aware masking. We formulate polynomial masking in terms of conditional affine observations. This characterizes the distribution exposed by a sequence of linear queries and distinguishes the dimension of an observation from its full affine support. We then give explicit low-degree interpolation spaces for prescribed Cartesian strata, witness slices, product trees, and scalar lookup sources. For two product trees whose joint interface reveals eight boundary evaluations and one root difference, our construction uses nine random coefficients and attains the dimension bound for independent uniform masking of that interface. All these source masks preserve the Boolean computation and have individual degree at most two.

An integrated zero-knowledge argument. We construct a multilinear argument combining query-aware source masks, division-free product-tree relations, a single global sumcheck, and a causal helper for its round messages. A typed pairing-based commitment scheme authenticates the permitted evaluations, including derived linear combinations, without adding separate root disclosures. We give an online simulator for the complete protocol and establish exact privacy under same-distribution simulation setup, ideal uniform private coins, and the specified classical proving interface. This privacy guarantee also accommodates an environment with quantum access to the unchanged challenge oracle. Classical knowledge soundness is established under a registered algebraic-group and random-oracle model, with degree bounds determined by the actual public SRS. The two security results are stated under their respective models.

Concrete costs and evaluation. We implement GoZK in Rust over BLS12-381 and evaluate its masking, sumcheck organization, and commitment costs. In the controlled eight-column configuration, query-aware masking uses 19 source coefficients, compared with 390 for an implemented projected arbitrary-point mask. It reduces proof length by 4,128 bytes and achieves paired prover-time improvements of 1.40–1.67 across three measured scales. Holding the masks fixed, one global sumcheck saves 6,752–10,400 bytes compared with seven native sumchecks. The native checks have slightly lower prover time at the two larger scales, making this primarily a communication improvement. Our repeated scale experiments reach 2^{20} witness cells, with a 21,877-byte proof at that size. We also derive exact communication formulas that explain how mask degrees, terminal values, and opening buckets contribute to the serialized proof.

1.3 Relation to Previous Work

Efficient zero knowledge for polynomial protocols has developed along several closely related lines. Zero-knowledge sumcheck and earlier polynomial-oracle constructions [14,27] establish how random polynomials can hide algebraic information while preserving the relation being checked. Libra [88] demonstrates that the masking polynomial can be much smaller than the polynomial underlying the computation. Its sumcheck mask uses randomness proportional to the number of messages exposed, while its GKR construction uses small low-degree masks for selected layer evaluations. GoZK builds on this principle but studies a different level of observation: the joint information disclosed by multiple committed sources, including linear combinations and observations conditioned on earlier messages.

HyperPlonk [26] provides both the arithmetization setting and another important precedent for structured masking. Its zero-knowledge compiler uses a vanishing factor in a selected coordinate and interpolates independent random values at query points whose values in that coordinate are distinct and non-Boolean. The compiler also adapts the challenge selection to enforce this restriction. GoZK instead develops masks for prescribed coordinate strata and joint linear observations. Its paired-tree construction, for example, does not require the two individual root evaluations to be independently uniform. This distinction permits a smaller mask for the actual root-difference interface while retaining

the low individual-degree bound. The broader sumcheck and commitment techniques remain part of the same multilinear proof-system lineage.

The choice of what a verifier may read already plays a central role in zero-knowledge PCPs. Kilian, Petrank, and Tardos construct proofs that hide the witness against a bounded number of queries [54]. Ishai, Mahmoody, and Sahai study how that bound interacts with efficient proof generation [52]. Gur, O'Connor, and Spooner use locally simulatable encodings in a masked sumcheck PCP for counting [47] and give constant-query PCPs for NP with perfect privacy against a prescribed adaptive query budget [48]. A related simulation principle appears in MPC-in-the-head, where the verifier receives selected parties' views [51]. Our source interface specifies the allowed linear observations as well as their number. In the paired-tree example, this distinction determines whether the two roots contribute one dimension or two.

The IOP model gives the verifier query access to successive prover messages [18]. This makes proof length and query complexity separate design parameters, as illustrated by constant-rate IOPs [15]. One line of work uses local tests for proximity to a Reed–Solomon code, from short PCPs [19] to FRI [12] and analyses of proximity gaps for affine families of received words [13]. GoZK authenticates its terminal values with a pairing-based opening. The rank computed in Section 3 concerns the joint distribution of those values.

These access patterns also distinguish implemented proof systems. Ligero uses coded MPC views [4], and Aurora constructs a transparent argument for R1CS from a univariate IOP [17]. Hyrax commits to the witness in a discrete-log group and proves claims about a layered circuit [85]. Virgo combines a layered circuit argument with transparent polynomial delegation [92]. Brakedown and Orion use linear-time encodable codes for their polynomial commitments [44,89]. Changing the commitment scheme changes the cost of opening each terminal point. The comparisons in Section 7 hold the pairing backend fixed.

A complementary line of work, exemplified by Spartan [74], develops efficient multilinear encodings and commitment-based succinct arguments for general computation relations. Our focus is different: given an indexed multilinear argument, we study how its zero-knowledge transformation can reflect the information that verification actually exposes. Work on polynomial-commitment simulation and extractability [59,73] also motivates treating the complete transcript as the object of the security proof. GoZK addresses this composition through an explicit conditional simulator and a separately specified classical extraction model.

Pinocchio encodes circuit satisfiability as a quadratic arithmetic program [66], and Groth gives a pairing-based argument with three proof elements [46]. Sonic supports a universal, updatable reference string of linear size [63]. Marlin combines such a setup with a holographic argument for rank-one constraint systems [28]. The arithmetization most directly relevant here is PLONK's use of a permutation to express wire-copy constraints across gates [41]. In GoZK the copy check builds two Boolean product trees. This exposes their boundary values and a single root difference, which determine the joint mask in Section 3.

Changing the constraint system also changes the polynomial checks available to the prover. Customizable constraint systems capture R1CS, Plonkish constraints, and

algebraic intermediate representations in a common format. Their SuperSpartan protocol extends Spartan's multilinear checks to that format [75]. Nova folds relaxed R1CS instances to verify successive steps of an incremental computation [56], and HyperNova folds multiple CCS instances [55]. GoZK combines the residuals of one indexed instance into a sumcheck. Each source then opens at the full points prescribed by that instance's gate, copy, and public-input checks.

The choice of lookup relation matters for the required openings. Plookup puts table entries and queried values in a common order and checks adjacent pairs [40]. PlonKup integrates that check with PLONK [67]. GoZK also uses adjacency, with multiplicities assigned to physical table positions and two explicit product trees. Logarithmic-derivative lookups certify multiplicities through a rational identity [49]. Caulk hides which positions of a committed table were used [90]. Caulk+ replaces a roots-of-unity subprotocol with a divisibility check, removing the table-size term from online prover complexity [69], and Baloo supports proofs about a committed subtable [91]. Lasso supports structured tables too large to materialize [76]. Jolt uses such tables to verify instruction evaluations [5]. Twist and Shout use one-hot addresses for read/write and read-only memory, respectively [77]. The scalar interface studied here fixes one explicit physical table.

Other implementations optimize where proof generation takes place. DIZK distributes proof generation and its memory load across a cluster [86]. The deVirgo prover in zkBridge distributes the checks for data-parallel circuits used in cross-chain verification [87]. Pianist distributes a PLONK-based prover [61], while HyperPianist distributes a multivariate proof protocol with logarithmic communication per machine [58]. The zkSpeed accelerator schedules HyperPlonk kernels, including sumcheck and multiscalar multiplication, on specialized hardware [31]. Section 7 varies the observation masks and the number of sumchecks on a fixed CPU backend. Several commitment schemes can authenticate polynomial evaluations. The DARK compiler uses groups of unknown order to instantiate polynomial IOPs [23]. Dory constructs a multivariate polynomial commitment from an argument for inner pairing products [57], and SLAP and Greyhound give polynomial commitments from lattice assumptions [2,64]. Pointproofs aggregates openings of positions across independent vector commitments [45]. GoZK batches openings at the same full point. Its mask dimension depends on the joint distribution of evaluations across all queried points.

The reference string also determines which polynomials an extractor can recover. Shoup studies discrete logarithms with generic group operations [80], and Bauer, Fuchsbauer, and Loss classify assumptions in the algebraic group model [9]. In our classical extraction experiment, a prover registers representations relative to the published group elements before the relevant oracle responses. We assume honest setup. Secure sampling of SNARK parameters, multiparty generation with a random beacon, and security under a subverted reference string each have their own setup experiments [16,21,1].

The random-oracle model of Bellare and Rogaway [11] permits a reduction to answer hash queries as they arrive. Pointcheval and Stern use forking in signature proofs [68]. For multi-round Fiat–Shamir protocols, the soundness loss can depend on the round structure and special soundness of the interactive proof [6,7]. Other constructions obtain

straight-line extraction [71] or instantiate Fiat–Shamir under computational assumptions for particular protocols [25]. Dao et al. show how omitting parts of a statement from a challenge hash can compromise soundness [32]. Our classical bound uses the statement and each round prefix encoded by the protocol.

Quantum access to the oracle changes what an extractor can learn from a query. Post-quantum Fiat–Shamir analyses give security conditions for the underlying interactive protocols [83,60,35]. Measure-and-reprogram techniques extend to multiple rounds [34], and online extraction applies when a classical output is tightly related to an oracle value [36]. Our privacy proof fixes the oracle and establishes equality of the external classical–quantum states. Our knowledge-soundness theorem assumes a classical prover in the registered algebraic experiment.

2 Relations, Observations, and Cost Parameters

We work over an odd prime field $\mathbb{F} = \mathbb{F}_q$. A witness table has $n = 2^\nu$ rows and $k = 2^c$ physical columns, hence $N = nk = 2^m$ cells with $m = \nu + c$. The core protocol works in dimension $L = m + 1$. The scalar lookup extension uses $L = \max(m + 1, \nu + 2)$, so the ambient Boolean cube has $M = 2^L = 2N$ points except in the single-column lookup case, where $M = 4N$. Coordinates are numbered from one in the mathematical presentation. Boolean tables in the implementation use most-significant-bit-first ordering and zero-based array indices.

For $x, y \in \mathbb{F}^d$, define the multilinear equality polynomial

$$\mathrm{eq}_d(x, y) = \prod_{i=1}^{d} ((1 - x_i)(1 - y_i) + x_i y_i).$$

The multilinear extension of a table $f : \{0,1\}^d \to \mathbb{F}$ is $\widetilde{f}(X) = \sum_{b \in \{0,1\}^d} f(b)\,\mathrm{eq}_d(X, b)$. We write $\phi(T) = T(T - 1)$. A mask divisible by a Boolean vanishing factor does not change the associated table on its Boolean domain. A polynomial's formal support, its restriction to the Boolean cube, and the coordinates published in a structured reference string (SRS) are distinct objects.

Indexed relation. An index fixes a gate expression, its public selector tables, a permutation of the N witness positions, and an optional public lookup table and query function. An instance supplies a public prefix of 2^{ν_p} witness cells, where $0 \le \nu_p \le m$. A witness satisfies the gate in each row, agrees across the specified permutation, and matches the public prefix. With lookup, each row's prescribed scalar query belongs to the physical public table. The implemented lookup functions are the sum of all columns and the first column. The index authenticates this choice before the prover supplies its witness.

Gate degree δ does not determine the complete gate evaluation cost. We separately record the number of monomials, selector handles, active columns, and the exponents actually evaluated. Physical columns that do not occur in the gate still participate in witness commitment and copy constraints. For the copy fingerprint, the position labels are injected into $\mathbb{F}$, requiring $q > N$. The scalar lookup construction assumes $q > 2n$

and $\nu \geq 1$. Challenge domains are chosen so that every required interpolation point and denominator is legal; in particular, sumcheck coordinates avoid $\{0, 1\}$.

Queries and source identities. A source is a registered polynomial together with its commitment and group type. A query consists of that source and its full L-coordinate point. Embedding a native relation into a suffix of the ambient variables happens before query deduplication. Two handles evaluated at the same point give two scalar claims, but their openings may share a bucket. An opening bucket is identified by both its source group and its full point. A derived source such as $A - B$ is registered as a linear combination: opening its value does not authorize separate disclosures of A and B at that point.

Pairing and commitments. Let $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ be asymmetric prime-order bilinear groups with generators G_1, G_2. Write $[a]_j = aG_j$ and $e([a]_1, [b]_2) = e(G_1, G_2)^{ab}$. Source commitments belong to $\mathbb{G}_2$; the global helper and opening messages use $\mathbb{G}_1$. The public SRS exposes evaluations at a secret point τ of specified polynomial spaces, as well as the hiding and pairing directions needed by the protocol. The actual spaces V and U include all public directions in the respective groups. They are not reduced merely because an honest source uses a smaller subspace. Section 4 specifies the commitment equations and the security interface.

Three degree and cost ledgers. For each sumcheck round a, E_a is the honest message cap and D_a bounds the residual allowed by the actual adversarial commitment spaces. A tighter E_a does not establish a tighter D_a. The compressed transcript sends $R_g = \sum_a E_a - 1$ round scalars. Let Q count transmitted endpoint scalars and let B count opening buckets, including the helper bucket. These are format quantities, not dimensions of the mask space. We use R_W for sampled witness-mask coefficients and explicitly distinguish successful field samples from random-byte requests.

Security models. The privacy experiment provides a classical proving interface and allows quantum queries to a fixed hash oracle [20]. We use the Fiat–Shamir transform [38] to derive each public challenge from the transcript up to that round. An honest simulator setup has the same public distribution as the real setup and retains its simulation trapdoor. Privacy is exact under the ideal private-coin distribution stated in Section 5. Knowledge soundness is a separate classical claim in the random oracle model with algebraic adversaries. Our representation interface builds on the AGM [39] and additionally requires online registration and restrictions on auxiliary inputs. Random evaluation for algebraic program testing and sparse polynomial computation goes back to DeMillo and Lipton [33] and Zippel [93]. The soundness analysis uses the polynomial identity bound of Schwartz [72] with the challenge distributions in Section 5. General background on multilinear extensions, sumcheck, and polynomial commitments can be found in [82].

3 Geometry of Masking

The source masks in GoZK preserve the Boolean tables used by the argument and hide the evaluations revealed during verification. Their construction depends on both the locations of these evaluations and the linear combinations in which they are disclosed. We begin with the two product trees used for the copy check, then develop the interpolation principle behind their masks. Throughout this section, $\phi(t) = t(t-1)$, and a query includes its complete evaluation point. We work over an odd prime field $\mathbb{F} = \mathbb{F}_q$ and number coordinates from one.

3.1 A running example with two product trees

Consider two product trees encoded by polynomials A and B. The protocol checks their leaves and parent–child relations, together with equality of the two products. At the end of the sumcheck, these checks require evaluations at non-Boolean points. Writing u, v for the first and last coordinates of a tree source, the parent, leaf, and two child queries have states $(0,*), (1,*), (*,0), (*,1)$, respectively. Here $*$ denotes an element of $\mathbb{F} \setminus \{0,1\}$. The remaining coordinates may differ across queries.

The root-equality check has state $(*,*)$ and requests $(A-B)(r)$ at a common complete point r. This terminal evaluation checks the root equation. The Boolean root address occurs inside the sumcheck. The joint interface therefore contains eight boundary evaluations and one difference. Masking all five evaluations of each tree independently would instead support ten independent answers. Our construction uses the nine-dimensional joint interface directly.

This example exhibits two features of the queries. Each source is evaluated once in each prescribed coordinate region, and the final observation combines two sources. Figure 1 shows the regions and the resulting observation matrix. All mask coefficients are sampled before the challenges that determine the evaluation points.

3.2 Conditional observations and interpolation

We first describe what a mask must hide after earlier observations have been revealed. Let R be a uniform vector of mask coefficients and s a fixed private state. At a reachable public history h, suppose the information about R is expressed by $A_h R = z_h - b_h(s)$. Assume that its conditional distribution is uniform on the nonempty solution set $r_{s,h} + K_h$, where $K_h = \ker A_h$. The next response is $Y = b(s,h) + L_h R$. The linear maps are determined by the public history, while the translations may depend on s.

Lemma 3.1 (Conditional affine observations). *Under these assumptions,*

$$Y \mid (s,h) \sim \mathrm{Unif}\big(b(s,h) + L_h r_{s,h} + L_h K_h\big). \tag{1}$$

Two private states compatible with h induce the same conditional response distribution exactly when these affine supports coincide. The dimension of the remaining observation space is

$$\dim L_h K_h = \operatorname{rank} \begin{bmatrix} A_h \\ L_h \end{bmatrix} - \operatorname{rank} A_h. \tag{2}$$

Proof. Every nonempty preimage of a response within $r_{s,h} + K_h$ is a coset of $K_h \cap \ker L_h$. These preimages have equal size, so the image distribution is uniform. Equality of the distributions is then equivalent to equality of their supports. The dimension formula follows from rank–nullity.

When a representative and a basis of the common affine support can be computed efficiently from h, the lemma also gives an exact sampler for the next response. The posterior assumption must account for every preceding message. Section 5.2 establishes it for the complete transcript, including commitments and openings. Appendix A.1 gives the general adaptive argument. Here we construct the source spaces that this argument uses.

A query region can be specified by a state $\sigma \in \{0, 1, *\}^d$. A Boolean entry fixes that coordinate, while a star requires a non-Boolean value. We call such a region a stratum. Let Σ be a collection of states, each with at least one star, and write $\ell_0(x) = 1 - x$, $\ell_1(x) = x$. Associate one polynomial with each state.

$$b_\sigma(X) = \prod_{i:\sigma_i=*} \phi(X_i) \prod_{i:\sigma_i\in\{0,1\}} \ell_{\sigma_i}(X_i), \qquad \mathcal{M}_\Sigma = \operatorname{span}\{b_\sigma : \sigma \in \Sigma\}. \tag{3}$$

Theorem 3.2 (Selected-stratum interpolation). *Choose one point x^σ in each selected stratum. Evaluation at these points is an isomorphism from $\mathcal{M}_\Sigma$ to $\mathbb{F}^\Sigma$. Every mask in this space vanishes on the Boolean cube and has individual degree at most two. Its dimension, $|\Sigma|$, is minimal among linear mask spaces that produce independent uniform values at these points.*

Proof. Order the states by their number of stars. An entry $b_\tau(x^\sigma)$ can be nonzero only if every starred coordinate of τ is starred in σ. Among states with equally many stars, distinct states have either a vanishing Boolean factor or a vanishing ϕ factor. The evaluation matrix is therefore triangular, with diagonal entry $\prod_{i:\sigma_i=*} \phi(x_i^\sigma) \neq 0$. This proves invertibility. The vanishing and degree properties follow from the basis, and surjectivity onto $\mathbb{F}^\Sigma$ requires at least $|\Sigma|$ dimensions.

The points may use different non-Boolean values, and the selected states need not satisfy a closure condition. There is one complete query per selected state. The mask has the same value at two points that differ only in ignored coordinates. Appendix A.2 gives the Cartesian-grid version.

3.3 A nine-coefficient mask for paired trees

For the four boundary states in Section 3.1, the selected-stratum space has the basis $\phi(v), u\phi(v), \phi(u), v\phi(u)$. The polynomial $\psi(u, v) = \phi(u)\phi(v)$ vanishes on these states and is nonzero at the terminal point of the root-equality check. One additional coefficient in either tree is therefore enough for the joint difference.

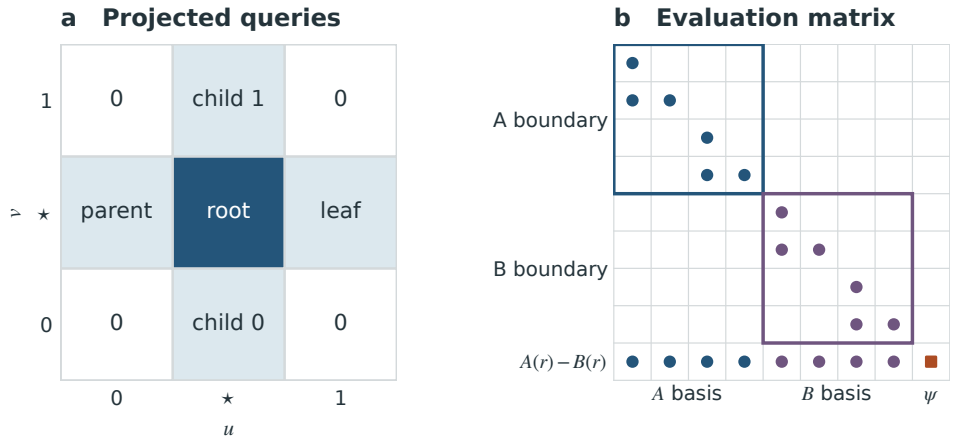

Fig. 1. One full query per stratum. The last row opens only $(A - B)(r)$ at the common full point r.

Construction 1. Joint Tree Masks

Input. The multilinear extensions A_0, B_0 of two tree tables and the joint query pattern, fixed before the challenges.

1. Sample $a_1, \ldots, a_5, b_1, \ldots, b_4$ independently and uniformly from $\mathbb{F}$.
2. Using the first and last native coordinates u, v, form
$$M_A = (a_1 + a_2 u)\phi(v) + (a_3 + a_4 v)\phi(u) + a_5\phi(u)\phi(v),$$
$$M_B = (b_1 + b_2 u)\phi(v) + (b_3 + b_4 v)\phi(u).$$
3. Output $A = A_0 + M_A$ and $B = B_0 + M_B$. The interface permits one evaluation in each boundary state for each source and the single value $(A - B)(r)$ at the prescribed common point.
4. Authenticate the difference using $C_A - C_B$. Reuse its value, or its negative, for subsequent requests.

Proposition 3.3 (Joint tree observations). *For any permitted choice of the query points, the nine coefficients in Construction 1 map bijectively to the eight boundary mask values and the mask difference at r. For fixed underlying tree tables, the nine masked observations are therefore jointly uniform. The construction has individual degree at most two and minimum dimension for this full uniform interface.*

Proof. For either tree, the parent observation determines its first coefficient. The leaf observation then determines the second, after subtracting the known contribution. The two child observations determine the third and fourth coefficients in the same way. Every divisor is nonzero by the prescribed coordinate states. The eight boundary observations thus determine eight coefficients without restricting a_5. The difference at the common point determines a_5, since $\phi(u)\phi(v) \neq 0$ at its first and last native coordinates. Bijectivity proves uniformity and the dimension bound. The degree bound follows from the construction.

3.4 Witness masks and related source spaces

Write the witness coordinates as (Z, C, X), with $c \geq 1$ column bits and $\nu \geq 1$ row bits. There are $k = 2^c$ columns. Extend the witness table by a public zero slice at $Z = 1$, and denote its multilinear extension by W_0. The committed source is $P_W = W_0 + M_W$. Its gate queries use Boolean column addresses and non-Boolean Z. The copy query has $Z = 0$ and non-Boolean column coordinates. The public-input query has both Z and the column coordinates non-Boolean.

Fix a column coordinate j before setup. Choose a uniform multilinear column polynomial $D(C)$ and independent uniform scalars d_0, d_1, and set

$$M_W(Z, C, X) = \phi(Z)D(C) + \phi(C_j)(d_0 + d_1 Z).$$

This mask has $k + 2$ coefficients. At the k gate points, the second term vanishes, so the mask values determine the Boolean table of D after division by the respective nonzero values of $\phi(Z)$. The copy observation then determines d_0. At the public-input point, subtracting the known terms leaves the nonzero multiplier $Z\phi(C_j)$ times d_1. Thus evaluation is an isomorphism onto the complete $k + 2$-value interface. The mask vanishes on the Boolean cube and has individual degree at most two.

Table 1 summarizes the source spaces used by the core and its scalar lookup extension. For lookup, the witness mask adds $\phi(X_\nu)D'(C)$, with an independent multilinear column polynomial D'. The additional k observations have $Z = 0$, Boolean column addresses, and non-Boolean X_ν. They determine D' before the preceding interpolation steps. The sequence source uses $\mathrm{span}\{\phi(Y_d), Y_1\phi(Y_d), \phi(Y_1)\}$, where $d = \nu + 1$. Its direct and two branch queries have states $(*, *), (0, *), (1, *)$ in the first and last coordinates.

With eight columns, the core sources use $10 + 9 = 19$ independent mask coefficients. Appendix A collects the extensions to general Cartesian grids, several trees, and low-dimensional interfaces. Using fewer gate columns restricts the observations and preserves their joint uniformity.

4 The GoZK Protocol

GoZK checks an indexed gate, a copy permutation, and a public-input slice. The default profile, core7, combines seven native residuals in one sumcheck. The witness and tree

Table 1. Source masks for the specified uniform observations.

Source	Independent observations	Dimension	Degree per axis
One tree	Four boundary values and one root-check value	5	≤ 2
Paired trees	Eight boundary values and one root difference	9	≤ 2
Core witness	k gate values, one copy value, one public-input value	$k + 2$	≤ 2
Lookup witness	Core observations and k lookup values	$2k + 2$	≤ 2
Lookup sequence	One direct value and two branch values	3	≤ 2

Witness counts use all k columns with $c \geq 1$. Lookup requires $\nu \geq 1$. Dimensions exclude helper coefficients and commitment blinds.

masks of Section 3 hide its terminal source observations. A separate polynomial masks its intermediate round messages. We describe the core for $c, \nu \geq 1$ and give the scalar lookup extension in Section 4.5.

4.1 Native relations and terminal observations

Write $n = 2^\nu$, $k = 2^c$, $N = nk = 2^m$, and $L = m + 1$. The index fixes the gate f, selector tables q, a permutation σ of the N positions, and a slice of length 2^{ν_p}. The statement supplies its values p. The masked witness $P_W(Z, C, X)$ restricts at $Z = 0$ to the Boolean witness table. After the witness commitment, the verifier samples $\beta \in \mathbb{F}_q^*$ and $\gamma \in \mathbb{F}_q$. With $\iota(I) = \sum_{a=1}^{m} 2^{m-a} I_a$ and s_σ the multilinear extension of $\iota \circ \sigma$, form the leaves

$$a(I) = P_W(0, I) + \beta\iota(I) + \gamma, \qquad b(I) = P_W(0, I) + \beta s_\sigma(I) + \gamma. \tag{4}$$

If the witness respects the copy permutation, reindexing its positions identifies the two products. Section 5 bounds fingerprint collisions for an invalid witness.

Encode each product tree on $\{0,1\}^{m+1}$, with leaves at $(1, I)$, root at $(0^m, 1)$, and unused address 0^{m+1}. For each $T = A, B$, compute the nonzero parents from the leaves upwards by

$$T(0, I) = T(I, 0)T(I, 1) \qquad (I \in \{0,1\}^m \setminus \{0^m\}). \tag{5}$$

This recurrence applies even when a leaf is zero. Apply the joint tree mask of Section 3 to the two multilinear extensions and commit independently to the resulting A, B. Set $\chi_{\mathrm{pub}}(Y) = \prod_{i=1}^{L-\nu_p}(1 - Y_i)$ and write $p_{\mathrm{suf}} = p(Y_{L-\nu_p+1:L})$ for the multilinear public input.

Column indices in the gate are fixed (c)-bit strings. Table 2 lists the private observations needed at the terminal point of each check. The two leaf checks share a witness value. Only the root check opens a linear combination of two committed trees.

Table 2. Native residuals and terminal observations in core7.

Check	Native residual	Private observations
Gate	$(1 - Z)f(q(X), P_W(Z, 0, X), \ldots, P_W(Z, k-1, X))$	One witness value per column
Copy leaf A	$A(1, I) - a(I)$	Leaf A, copy witness
Copy leaf B	$B(1, I) - b(I)$	Leaf B, copy witness
Internal A, B	$(1 - \mathrm{eq}(I, 0^m))(T(0, I) - T(I, 0)T(I, 1))$, once for each tree T	Parent and both children of each tree
Root equality	$\mathrm{eq}(Y_{1:m}, 0^m)Y_L(A - B)(Y)$	One difference at Y
Public input	$\chi_{\mathrm{pub}}(Y)(P_W(Y) - p_{\mathrm{suf}})$	Witness at Y

All residuals in Table 2 vanish on their Boolean domains for a valid witness. The root residual has a single possible nonzero Boolean entry, so its Boolean sum equals the difference of the two products. We test this sum directly. Each of the other six checks uses a random equality weight to test pointwise vanishing.

4.2 Causal masking of one global sumcheck

For the residual R_j in m_j variables, set $F_j(x) = R_j(x)\,\mathrm{eq}(x, \omega_j)$, except for a canonical root where $F_j = R_j$. Following the linear combination of checks in HyperPlonk [26], align the variables on common suffixes and set

$$\widetilde{F}_j(Y) = \mathrm{eq}(Y_{1:L-m_j}, 0^{L-m_j}) F_j(Y_{L-m_j+1:L}), \tag{6}$$

$$F_*(Y) = \sum_{j=1}^{J} \lambda_j \widetilde{F}_j(Y), \qquad \lambda_1 = 1. \tag{7}$$

The prefix equality factor has Boolean sum one. Hence the embedding preserves every native sum. At the terminal vector r, each relation uses $r_{L-m_j+1:L}$. A gate column b reads $P_W(r_{c+1}, b, r_{c+2:L})$. The leaf checks read $P_W(0, r_{2:L})$ and $T(1, r_{2:L})$. The internal checks read $T(0, r_{2:L})$, $T(r_{2:L}, 0)$, and $T(r_{2:L}, 1)$. The public and root checks read $P_W(r)$ and $(A - B)(r)$. Indexed evaluations follow their own complete point maps in Appendix C.

For the intermediate sums we use the univariate-mask approach of Libra [88]. For fixed message caps $E_1 \geq 2$, $E_a \geq 1$ for $a > 1$, draw independent uniform field elements $\alpha_e, c_{a,e}$ and set

$$g(Y) = \sum_{e=2}^{E_1} \alpha_e (Y_1^e - Y_1) + \sum_{a=2}^{L} \sum_{e=1}^{E_a} c_{a,e} (Y_a^e - \tfrac{1}{2}). \tag{8}$$

The first block vanishes at both endpoints. Each later univariate term has opposite values at zero and one. Consequently

$$\sum_{b \in \{0,1\}^{L-1}} g(0, b) = \sum_{b \in \{0,1\}^{L-1}} g(1, b) = 0. \tag{9}$$

Proposition 4.1 (Causal round masking). *Fix the source state and pre-round public transcript. Suppose the helper coefficients are independent uniform under this conditioning. At each reached history of the sumcheck on $F_* + \rho g$, for $\rho \neq 0$, the first-round coefficients of degrees $2, \ldots, E_1$ are jointly uniform. In round $a > 1$ the coefficients of degrees $1, \ldots, E_a$ are jointly uniform. Challenges may depend on the preceding messages.*

Proof. Summation over a future block's Boolean coordinate cancels that entire block in every earlier message. Past blocks contribute only constants. The current free coefficients therefore equal fixed source coefficients plus the fresh uniform block scaled by $\rho 2^{L-a} \neq 0$. This holds after conditioning on each preceding round in order.

Figure 2 shows where each block can first affect a round message. Independent commitment blinds preserve the pre-round condition in the complete protocol (Section 5).

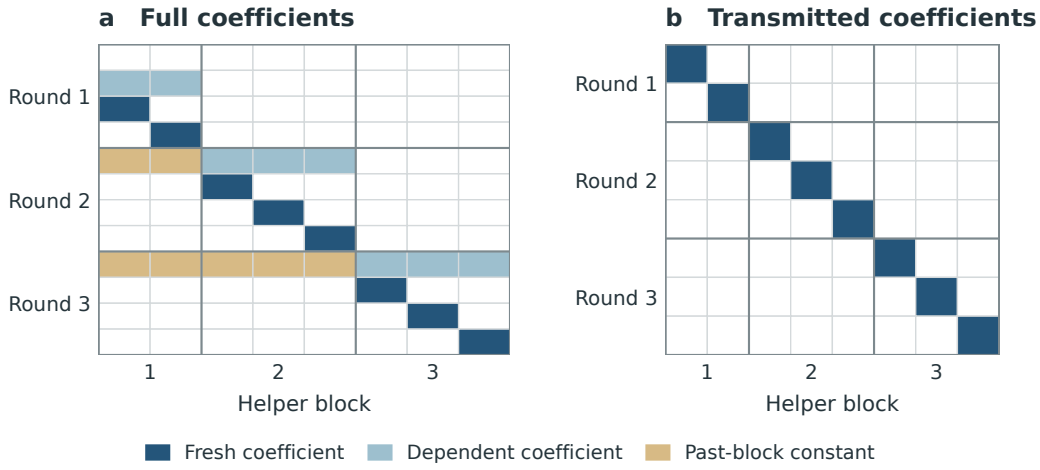

Fig. 2. Helper contributions to full and transmitted round coefficients ($L = 3$, $E = (3, 3, 3)$).

Protocol 1. Causal Masked Global Sumcheck

Common input. Source commitments, certified native checks, coordinate maps, and caps E. The prover knows the committed sources.

1. The prover samples g by (8) and sends its independently blinded type-1 commitment C_g.
2. The verifier samples one packet containing independent $\omega_j \in \mathbb{F}_q^{m_j}$ for each non-root check, $\lambda_2, \dots, \lambda_J \in \mathbb{F}_q$, and $\rho \in \mathbb{F}_q^*$. Set $v_0 = 0$ and $P = F_* + \rho g$.
3. In round $a = 1, \dots, L$, the prover forms $h_a(T) = \sum_{b \in \{0,1\}^{L-a}} P(r_{1:a-1}, T, b)$. In the first round send coefficients $h_{1,e}$ for $e = 2, \dots, E_1$. In every later round send $h_{a,e}$ for $e = 1, \dots, E_a$. Reconstruct the omitted coefficients by (10) and (11).
4. After the message of round a is fixed, the verifier samples $r_a \in \mathbb{F}_q \setminus \{0, 1\}$. Both parties set $v_a = h_a(r_a)$ and repeat the preceding step for the next axis.
5. The prover supplies each prescribed source observation once, including the registered index values. The verifier evaluates $F_*(r)$ and returns the source claims and (C_g, r, y_g) with $y_g = (v_L - F_*(r))/\rho$ for authentication.

More generally, if the two honest first-coordinate half sums are public s_0, s_1, the first codec below applies. GoZK has $s_0 = s_1 = 0$. With $h_a(T) = \sum_e h_{a,e} T^e$, the omitted coefficients are

$$h_{1,0} = s_0, \qquad h_{1,1} = s_1 - s_0 - \sum_{e=2}^{E_1} h_{1,e}, \tag{10}$$

$$h_{a,0} = \tfrac{1}{2}\left(v_{a-1} - \sum_{e=1}^{E_a} h_{a,e}\right) \quad (a > 1). \tag{11}$$

The recurrence and final opening enforce

$$h_a(0) + h_a(1) = v_{a-1}, \qquad v_a = h_a(r_a), \tag{12}$$

$$y_g = (v_L - F_*(r))/\rho, \tag{13}$$

$$v_L = F_*(r) + \rho g(r). \tag{14}$$

The rounds transmit $\sum_a E_a - 1$ scalars. The Boolean sum of g is fixed and y_g is derived, so neither needs a separate scalar.

4.3 Commitments and authentication of the observations

Use prime-order pairing groups $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$, with $[v]_t = vg_t$. Let V span the ambient multilinears and the registered source-mask directions. Write $Q(V)$ for their coordinate-order telescoping quotient space and set

$$U = Q(V) + \mathrm{span}\{Y_a^e : 1 \le a \le L,\ 1 \le e \le E_a\}. \tag{15}$$

Setup samples $\tau \in \mathbb{F}_q^L, \zeta \in \mathbb{F}_q^*$ and publishes basis evaluations of V in $\mathbb{G}_2$ and U in $\mathbb{G}_1$, the generators, $[\tau_a]_2$, and both $[\zeta]_t$. Commit to P as $C_P = [P(\tau) + r_P\zeta]_t$, with a fresh uniform private blind or a zero public index blind. Sources and index handles have type 2 and the helper has type 1.

At a complete point z, write $P(Y) - P(z) = \sum_a (Y_a - z_a)Q_a(Y)$. With independent uniform ξ_a, send

$$\Pi_a = [Q_a(\tau) + \xi_a\zeta]_1, \qquad \Pi_0 = [r_P - \sum_a \xi_a(\tau_a - z_a)]_1. \tag{16}$$

Appendix C gives the public computation of both terms. For $\mathrm{Left}_1(C, y) = e(C - [y]_1, g_2)$ and $\mathrm{Left}_2(C, y) = e(g_1, C - [y]_2)$, check

$$\mathrm{Left}_t(C_P, y) = e(\Pi_0, [\zeta]_2) \prod_{a=1}^{L} e(\Pi_a, [\tau_a - z_a]_2). \tag{17}$$

Telescoping cancels the fresh blinds. Each opening uses $L + 1$ elements of $\mathbb{G}_1$. Claims share an opening precisely when their group and complete L-coordinate point agree. After all values are fixed, the verifier samples coefficients $(1, \alpha_2, \ldots)$ in each bucket. The derived handle $C_A - C_B$ authenticates $(A - B)(r)$ using the corresponding differences of polynomials and blinds.

For soundness, f_a bounds the assembled expression degree on axis a over the full source-group span, including equality and prefix factors. Let u_a bound the degree allowed by the actual helper-group span. Set

$$D_a = \max\{E_a, f_a, u_a\}. \tag{18}$$

The profile fixes E, D before proving. Honest encodings use E, and Section 5 charges D for a false sumcheck claim.

4.4 The complete core protocol

Protocol 2. The Complete GoZK Core Protocol

Common input. An indexed statement and its core7 parameters. The prover knows a witness.

1. The prover masks the witness and sends C_W.
2. The verifier samples (β, γ). The prover constructs the leaves in (4), builds and masks the trees, and sends C_A, C_B.
3. Run Protocol 1, obtaining the helper commitment, the sumcheck transcript, and its terminal claims.
4. Fix all source values, the derived root and helper claims, and the ordered bucket list. Draw the same-point weights. The prover sends one freshly randomized opening per bucket.
5. Accept if the proof decodes completely, every round satisfies its codec and recurrence, and the openings verify against the terminal claims.

The Fiat–Shamir transcript binds the profile, index, statement, proof identifier, and preceding messages at each challenge. The verifier may combine the opening equations after fixing the complete openings, using an additional challenge packet as specified in Appendix C.

4.5 Scalar lookup extension

For a fixed table of $n-1$ physical entries, assign each of the n scalar queries to a matching entry. If a_j queries use position j, write $1 + a_j$ copies of t_j in physical order. A masked source P_H stores the resulting $2n - 1$ entries along a public cycle. A public factor sets its semantic zero address to zero, and its commitment precedes the lookup fingerprint. Two additional product trees compare the query and table transitions with those of the repeated sequence. Four residuals check their leaves and internal nodes, and one checks the new root difference. The extension uses twelve checks with core7, or eleven with the virtual public-input profile. It runs the same sumcheck in $L = \max(m + 1, \nu + 2)$ coordinates. Protocol C.2 in Appendix C specifies all five residuals, the scalar fingerprints, the physical addresses, and the opening points.

5 Security

The protocol is perfectly complete. Its privacy follows from an online simulation of the conditional message distributions, while knowledge soundness follows by extracting the committed witness and tracing an invalid relation through the checks of Section 4.

5.1 Completeness and simulation setup

Fix a core or single-table scalar-lookup profile specified in Section 4 and Appendix C. Assume that q is an odd prime, $q > N$, and $q > \max_a D_a + 2$. Lookup additionally requires $q > 2n$.

Theorem 5.1 (Perfect completeness). *For every valid indexed statement and witness, the protocol accepts for every permitted challenge sequence and every choice of honest masks and blinds.*

Proof. The source masks preserve all Boolean table entries, and virtual public slices have their specified values identically. The leaf and internal relations reconstruct the two products, and their roots agree for a valid copy assignment, including when products are zero. In the lookup extension, each repeated table entry contributes its original cycle edge and one self-loop per assigned query, giving the same product on both sides. Thus all native residuals vanish on their Boolean domains, and multiplication by the zero-prefix selector preserves this pointwise vanishing. The helper satisfies the two first-round endpoint constraints and the subsequent sumcheck recurrences. Its derived terminal value is its actual evaluation. Finally, telescoping gives every opening equation.

For privacy, SimSetup samples the same (τ, ζ) as real setup and retains them. Its public parameters have exactly the real setup distribution. The simulator receives the public index polynomials and their prescribed representations. It creates all private commitments in the simulated sessions and records their exponents.

5.2 Exact causal privacy

The privacy experiment exposes classical proof requests for valid indexed statements and the resulting public protocol messages. Each request starts a fresh session, which advances through the prescribed state machine and uses independent uniform private field coins, including fresh randomness for every opening. The environment may interleave sessions, hold quantum auxiliary information correlated with the classical witnesses, and query the challenge function coherently. Source observations follow the interfaces of Section 3 and Appendix A. Requests and messages are classical. A session advances once per transition with fresh coins, and the prover's private state is inaccessible. The interface excludes resets, active corruption, and coherent proof requests.

Algorithm 1. Exact Online Simulator

Input. The profile, public index polynomials and statements, simulation trapdoor (τ, ζ), and oracle access to the original challenge function H.
State. Keep the transcript, an exponent for each logical commitment handle, and a joint observation cache for each session. Identify each observation by the proof, source handles, canonical linear functional, and complete ambient point. Initialize the public exponents by evaluating the index polynomials at τ. Sample the protocol salt as in the real prover. Use the prescribed transcript context at every challenge stage.

1. **Commitments.** For a private type-t commitment, sample $c \leftarrow \mathbb{F}$, output $[c]_t$, and store c. Compute each derived handle as the prescribed linear combination of its parents. Keep separate logical handles even when their encodings coincide.
2. **Rounds.** Sample uniform free coefficients of degrees $2, \ldots, E_1$ in the first round and $1, \ldots, E_a$ subsequently. Recover the constrained coefficients as in Protocol 1, append the message, and obtain the next challenge from H.
3. **Source observations.** Compute a representative v and basis matrix M of the common conditional support from the cache and the interpolation maps in Section 3. Return $v + M\eta$ for uniform coordinates η and record the new constraints. Use stored values for dependent observations and the public index polynomials for indexed evaluations. Store the root difference as a single linear observation. Derive $y_g = (v_L - F^*(r))/\rho$ after fixing the terminal values.
4. **Batching.** Freeze the terminal values, derived claims, and canonical bucket list. Obtain the same-point weights from H and combine the stored commitment exponents.
5. **Openings.** For a bucket with exponent c, point z, and value y, sample fresh independent $a_1, \ldots, a_L \in \mathbb{F}$. Set
$$a_0 = \zeta^{-1} \left(c - y - \sum_{i=1}^{L} (\tau_i - z_i) a_i \right).$$
Send $[a_1]_1, \ldots, [a_L]_1$ followed by $[a_0]_1$, in the real wire order. Obtain optional post-opening aggregation weights after the entire opening transcript is fixed.

Theorem 5.2 (Exact causal privacy). *Fix any classical function H, defined on the full oracle domain, whose decoders return legal challenges. Under the stated classical interface, ideal independent uniform private coins, and same-distribution* SimSetup*, Algorithm 1 efficiently simulates the protocol without the witnesses and without changing H. At every public-message boundary, its external classical–quantum state equals the real external state. Averaging over a random H therefore gives zero distinguishing error against an environment with quantum access to that oracle.*

Proof (Proof sketch). For every fixed polynomial, its commitment exponent $P(\tau) + \zeta r_P$ is uniform since $\zeta \neq 0$. Conditioning on the commitment determines its blind by a bijection. Proposition 4.1 gives the same uniform free-coefficient law as the round handler. For any fixed source state, the consumed helper coefficients can be reconstructed from the round messages. The resulting bijection leaves the source-mask posterior unchanged. The interpolation results therefore give the common conditional support used by the source handler, including the single root difference and its dependent repetitions. Finally, the honest opening exponents are uniform on the affine hyperplane sampled by the opening handler. Each $Q_i(\tau) + \zeta \xi_i$ is uniform, and telescoping determines a_0.

These conditional laws agree within each fixed classical witness branch. Between responses, the environment applies the same quantum operations and the same oracle calls to equal external states. Induction over the reached message histories proves the claim. Appendix B.2 gives the branchwise invariant and the complete composition argument, including interleaved sessions.

5.3 Classical knowledge soundness

Knowledge soundness uses a registered online variant of the algebraic group model [39], with a classical adversary and a random oracle. Each newly created G_1 or G_2 handle receives a polynomial-size algebraic representation, normalizable to the actual public SRS lists when it first appears in a parsable Fiat–Shamir query. That representation is frozen before the query is answered. Final proof elements are also registered. Each handle keeps its context and interpretation throughout the execution. Public index handles retain the indexer's interpretation, and derived handles retain their prescribed parent combinations.

Adversarial index choices pass through the prescribed indexer. Auxiliary group inputs have representations available to the reduction. Auxiliary scalars and byte strings are generated by a reduction-runnable procedure from the public parameters and an initial state independent of the setup secrets, or their joint distribution is explicitly covered by the hardness assumption.

Let $b_U, b_V \geq 1$ bound the total degrees of the actual published spaces $\mathcal{U}, \mathcal{V}$, and put $\Delta = \max\{b_V, b_U + 1\}$. The computational assumption is the (b_U, b_V)-power discrete-log problem. Given $[a^j]_1$ for $0 \leq j \leq b_U$ and $[a^j]_2$ for $0 \leq j \leq b_V$, recover uniform unknown $a \in \mathbb{F}_q$. Appendix B.3 bounds the probability of any accepted opening disagreeing with its frozen polynomial by

$$\epsilon_{\mathrm{PCS}} \leq \mathrm{Adv}^{\mathrm{powDL}}_{b_U, b_V}(\mathcal{B}) + \Delta/q + \epsilon_{\mathrm{alg}}. \tag{19}$$

The bound covers all openings in one evaluation-consistency event. The algebraic-subroutine term is zero for the expected-polynomial-time reduction. A strict cutoff retains the explicit failure bound given there.

For a challenge decoder with M outputs, write μ_M for its largest output probability. Uniform decoding gives $\mu_M = 1/M$. Reduction of b uniform bits modulo M gives $\mu_M = \lceil 2^b/M \rceil / 2^b$. Coordinates of a fresh packet use independent raw randomness. The fingerprint degree bound is $d_{\mathrm{fp}} = \max\{N-2, 0\}$ for the core and $d_{\mathrm{fp}} = \max\{N-2, 2n-1\}$ with lookup. Its error is at most

$$\theta_{\mathrm{fp}} = d_{\mathrm{fp}} \max\{\mu_q, \mu_{q-1}\}. \tag{20}$$

The equality and mixing packet has error at most $\eta_{\mathrm{pkt}} = \max\{1 - (1-\mu_q)^{h+1}, \mu_{q-1}\}$, where h is the largest native dimension of an equality-weighted residual. A sumcheck transition has error at most $D_a \mu_{q-2}$, using the actual degree bound from Section 4. Same-point batching has error at most $\theta_{\mathrm{batch}} = \mu_q$, or zero when all buckets are singletons. These are conditional bounds at a reached transcript history.

Let $Q_{\mathrm{fp}}, Q_{\mathrm{pkt}}, Q_a, Q_{\mathrm{batch}}$ bound distinct oracle queries at the corresponding stages, including abandoned branches, preprocessing, and permitted auxiliary computations. Put

$$\epsilon_0 = \mathrm{Adv}^{\mathrm{powDL}}_{b_U, b_V}(\mathcal{B}) + \Delta/q + \epsilon_{\mathrm{alg}} + \epsilon_{\mathrm{ctx}} + \epsilon_{\mathrm{chain}}. \tag{21}$$

The last two terms cover collisions and prebound references in compressed contexts and transcript chains. Both vanish for an injective full-context encoding. Appendix B.6 specifies their bounds for the stated digest construction.

Theorem 5.3 (Classical knowledge soundness). *Under honest setup, the registered experiment and input conditions above, every classical polynomial-time adversary admits a non-rewinding polynomial-time extractor, with output w_{ext}, and a power-discrete-log reduction such that*

$$\begin{aligned} &\Pr[\mathsf{Verify}(i, x, \pi) = 1 \ \wedge \ R(i, x; w_{\mathrm{ext}}) = 0] \\ &\quad \le \epsilon_0 + (Q_{\mathrm{fp}} + 1)\theta_{\mathrm{fp}} + (Q_{\mathrm{pkt}} + 1)\eta_{\mathrm{pkt}} \\ &\qquad + \mu_{q-2} \sum_{a=1}^{L} (Q_a + 1) D_a + (Q_{\mathrm{batch}} + 1)\theta_{\mathrm{batch}}. \end{aligned} \tag{22}$$

The bound is capped at one. Each additional trial covers a query made for the first time during final verification.

Proof (Proof sketch). The extractor evaluates the frozen witness source, with the profile's public-input substitution if present, on its designated Boolean slice. An invalid extracted relation yields a fingerprint defect or a nonzero native residual. The joint-packet bound covers both a residual hidden by compression and an adversarial helper with nonzero total. A false sumcheck claim can become correct at round a only at a root of a nonzero polynomial of degree at most D_a. At the endpoint, a surviving false claim gives an incorrect source observation or derived helper value. Same-point batching either preserves an incorrect claim or cancels it with the stated probability. Accepting the former gives the shared evaluation-consistency event.

Each bad set is fixed before its fresh oracle answer. The local bounds therefore apply at every reached history. Summing over the stage budgets, together with the shared evaluation-consistency and context events, gives the displayed bound. Appendix B gives the complete reduction and the sharper survival-product bound.

If verification aggregates the opening equations through a separate Fiat–Shamir stage, add $(Q_{\mathrm{pair}} + 1)\mu_q$ to the bound. With private uniform verifier weights sampled after the complete openings, the additional loss is at most $1/q$. The ordinary soundness bound follows since every witness for a false statement is invalid.

6 Implementation and Concrete Costs

The Rust prover, verifier, and tests are available at `https://github.com/cxcx52/GoZK`. The implementation uses arkworks over BLS12-381. It reconstructs the coordinate maps,

source spaces, group types, and degree caps from the profile and checks the indexed public polynomials against the authenticated SRS. It samples private field elements by rejection from OsRng and decodes challenge packets from SHAKE256 output.

6.1 Native tables and reusable commitments

The prover stores Boolean tables separately from their low-degree mask terms. Each source occurrence has a map of fixed, variable, or complemented coordinates. These maps let the prover fold a table along the challenges of its native suffix and evaluate sparse mask terms alongside the fold. This avoids materializing a globally padded table for each residual.

Two computations reuse public group elements. To derive low-degree monomial commitments, the loader marginalizes the Boolean Lagrange basis over unused coordinates and applies a superset zeta transform. For the nondegenerate core, deriving this cache costs $O(N + k(c+1))$ additions and $O(k+L)$ temporary group elements, in addition to the dense tables and SRS. To open an immutable dense table, the prover reuses its unblinded quotient commitment when the table, SRS, embedding, and preceding opening coordinates agree. Fresh independent blinds still randomize all $L+1$ elements sent per opening.

Earlier implementations of streaming interactive proofs separate the verifier's access to the input from the prover's work on the arithmetic circuit [30,29]. Thaler obtains a time-optimal circuit prover for suitable wiring functions [81], and Vu et al. use circuit structure to select a verification protocol [84]. The field and the representation of the input tables also matter. Small-characteristic sumcheck can move many multiplications into a base field [8], while time–space tradeoffs govern how much of a folded table the prover retains [10]. Here the tables are over the BLS12-381 scalar field and the measured prover retains their successive folds.

At fixed width and bounded per-entry expression cost, geometric folding uses linear work in the table size. Gate evaluation, public-input processing, MSMs, and quotient computation also enter the complete cost. Appendix C gives the basis, cache, and operation counts with their parameter dependence.

6.2 Communication from the observation interface

A field element occupies 32 bytes, a $\mathbb{G}_1$ point 48, and a $\mathbb{G}_2$ point 96. The five-byte header and 32-byte salt give 37 fixed bytes. For the one-global-sumcheck format, let R_g count round scalars, Q transmitted endpoint scalars, and B complete-point opening buckets. The proof length is

$$|\pi| = 37 + C + 32(R_g + Q) + 48B(L+1), \qquad R_g = \sum_a E_a - 1. \tag{23}$$

The commitment block is $C = 3\cdot 96 + 48 = 336$ bytes for the core and $C = 6\cdot 96 + 48 = 624$ with lookup. Index commitments belong to the public parameters, but the prover's values

for their evaluations count in Q. The helper contributes a type-1 bucket and its value is derived.

For $c, \nu \geq 1$, let k_{act} count active gate columns and s count distinct committed selector handles. The full points in Appendix C.3 give

$$Q_7 = k_{\text{act}} + s + 12, \qquad Q_6 = k_{\text{act}} + s + 11, \qquad B_7 = B_6 = k_{\text{act}} + 7. \tag{24}$$

There are $k_{\text{act}} + 6$ distinct source-group points on every permitted core challenge path. The helper opens at the same coordinates as the root difference and occupies the other group.

For the thirteen-selector gate with eight active columns, core7 has $Q = 33$ and $B = 15$. At degree five, anchor three, $\nu_p = \nu$, and the tight honest caps of the scale family, $L = m + 1$ and $R_g = 7m + 4$ for $m = \log_2 N \geq 4$. Thus

$$|\pi| = 944m + 2997. \tag{25}$$

An extra axis adds seven coefficients and one group element in each of fifteen openings, or $7 \cdot 32 + 15 \cdot 48 = 944$ bytes. The QG reference in the mechanism study uses $E = D$, which raises the last cap from seven to twelve. Its proof uses $R_g = 7m + 9$ scalars in the rounds and $944m + 3157$ bytes in total. At $N = 2^{16}$ the two lengths are 18,101 and 18,261 bytes, respectively.

6.3 Fewer relations can require more bytes

The core6 profile incorporates the public slice into each witness occurrence through (42). This saves one terminal evaluation. The substitution also multiplies witness expressions by a public prefix factor, which can increase their honest round caps.

At $N = 1024$, $k = 8$, degree five, $\nu_p = 1$, anchor three, and tight honest caps, the thirteen-selector gate uses 74 round scalars in core7 and 108 in core6. The endpoint count falls from 33 to 32 and both profiles use 15 openings. Figure 3 accounts for all 1,056 additional bytes in $12{,}437 + 34 \cdot 32 - 32 = 13{,}493$.

7 Evaluation

We compare the default GoZK protocol with three controls that change source masking, sumcheck organization, or the first-round helper constraint. A separate scale study measures end-to-end performance on five witness sizes. The engineering experiments measure quotient caching and the scalar lookup extension.

7.1 Experimental design

Table 3 defines the four variants. The mechanism study fixes the thirteen-selector gate, eight columns, degree five, the public slice $\nu_p = \nu$, anchor three, and cached openings. At each of $N = 2^{10}, 2^{14}, 2^{16}$, ten numerical witnesses form ten matched blocks. QG, GG,

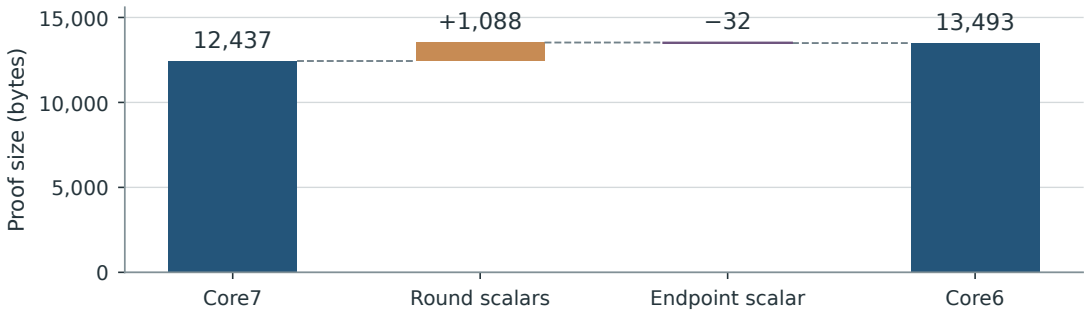

Fig. 3. Public-prefix compilation at $N = 1024$, $k = 8$, $\nu_p = 1$, anchor three, and tight caps for the degree-five fixed 13-selector gate.

and QM run once per block, each in a fresh process on four physical cores of one Xeon Gold 6230 socket. The scheduler interleaves the scales and randomizes arm order, with each alternative preceding QG in five blocks per scale. QM shares QG's certified index and SRS. GG uses the same terminal source observations and bucket count. A second randomized experiment compares QG and QT on the same 30 witnesses using the three original QG keys, with identical caps and opening structure.

Table 3. Protocol variants in the two matched mechanism experiments.

Arm	Source masks	Sumchecks	Change relative to QG
QG	Query-aware	One global	Reference
GG	Projected arbitrary-point	One global	Recertify the source and quotient supports and degree caps
QM	Query-aware	Seven native	Use seven independent helpers and their required openings
QT	Query-aware	One global	Send one additional first-round coefficient

The scale and engineering studies use an Intel Xeon E5-2696 v4 at 2.20 GHz under an 18-CPU, 70-GiB quota. Processes run on physical cores of one socket in blocked order and reuse a checked SRS and index across inputs. Both platforms use Rust, arkworks, and BLS12-381 with fresh private randomness per proof.

Online proving includes encoding. Verification starts from bytes and includes decoding, subgroup checks, and the cryptographic equations. SRS loading, indexing, and input generation are timed separately. Process peaks include initialization and verification. We report medians and 95% percentile-bootstrap intervals from 10,000 resamples. For paired comparisons we form a ratio within each witness and resample the witness pairs. These intervals describe sampling variation within the recorded data. All 150 mechanism-study proofs passed both individual and aggregate verification and subsequently passed both checks in fresh-process replay. Appendix D gives the timing boundaries and records. Appendix B.8 gives the security instantiations of the controls.

7.2 Effects of the masking and sumcheck mechanisms

Figure 4 gives the prover-time ratios for the three controls and the proof-size components at $N = 2^{16}$.

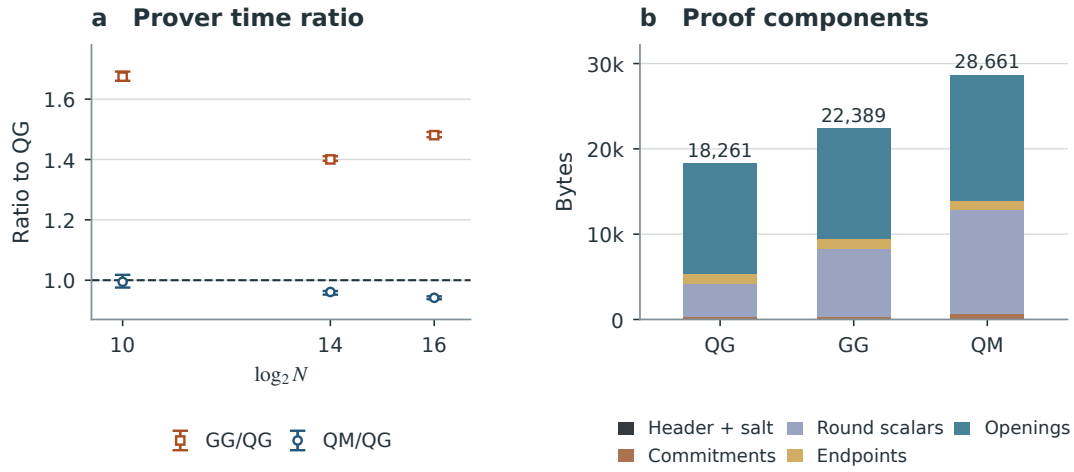

Fig. 4. Matched prover-time ratios with 95% intervals (left) and proof components at $N = 2^{16}$ (right), Xeon Gold 6230.

Source masking. GG supports arbitrary eligible points on each registered projection. Its witness and two tree masks use $348 + 21 + 21 = 390$ coefficients. QG masks the actual gate, copy and public observations, including the single joint root difference, with $10 + 5 + 4 = 19$ coefficients. These counts cover the source masks alone.

The median paired GG/QG prover-time ratios are 1.674, 1.400, and 1.481 at the three increasing scales. Both protocols transmit 33 endpoint scalars and use 15 opening buckets. GG's larger caps add 129 round scalars and account for its full $129 \cdot 32 = 4{,}128$-byte proof increment. Its SRS contains 350 additional $\mathbb{G}_2$ elements and 337 additional $\mathbb{G}_1$ elements, or 49,776 bytes. At $N = 2^{16}$ the median process peak is smaller for GG, 146.4 MiB compared with 162.5 MiB for QG.

The norm–trace construction in Appendix A.7 uses 55 coefficients on the same projections. Its witness and tree total-degree bounds are 17 and 8, compared with 11 and 6 for GG. This alternative contributes an analytical entropy–degree comparison. Its proving time and serialized SRS size were not measured.

Global aggregation. QM runs seven native sumchecks on the QG masks and shares their challenges on aligned suffixes. Each check has a separately blinded helper and a derived helper endpoint. At the three scales, QG saves 6,752, 9,184, and 10,400 bytes. At $N = 2^{16}$ the 262 additional round scalars, six additional helper commitments, and two additional opening buckets in QM give the difference

$$262 \cdot 32 + 6 \cdot 48 + 2 \cdot 18 \cdot 48 = 10{,}400 \text{ bytes.}$$

The paired QM/QG prover-time ratios are 0.995 $[0.976, 1.018]$, 0.961 $[0.952, 0.964]$, and 0.942 $[0.937, 0.947]$. The smallest-scale interval includes one. At the two larger scales, the seven native checks take approximately 4% and 6% less prover time while sending longer proofs.

First-round constraint. QT adds an independent term $a(Y_1 - 1/2)$ to the helper. Its Boolean total remains zero and its first-coordinate half sums can differ. The first-round linear coefficient is transmitted, and the constant follows from the total. Both arms derive the terminal helper evaluation and open it at the same point type. Across all 30 pairs QG saves exactly one private field sample and 32 bytes, with the same degree caps and opening counts. The QT/QG time ratios are 1.015 $[0.988, 1.030]$, 1.000 $[0.995, 1.009]$, and 0.999 $[0.990, 1.004]$. All three time and peak-memory ratio intervals include one. The data resolve the one-scalar communication saving and leave the timing effect unresolved at these scales.

7.3 End-to-end scale

Figure 5 shows the five completed E5 scale points using four threads, the thirteen-selector gate, $k = 8$, degree five, $\nu_p = \nu$, core7, anchor three, cached openings, and tight honest caps. There are ten proofs at each of $N = 2^{12}, 2^{14}, 2^{16}, 2^{18}$ and five at $N = 2^{20}$. The largest witness has 2^{20} cells across $2^{17} = 131{,}072$ rows.

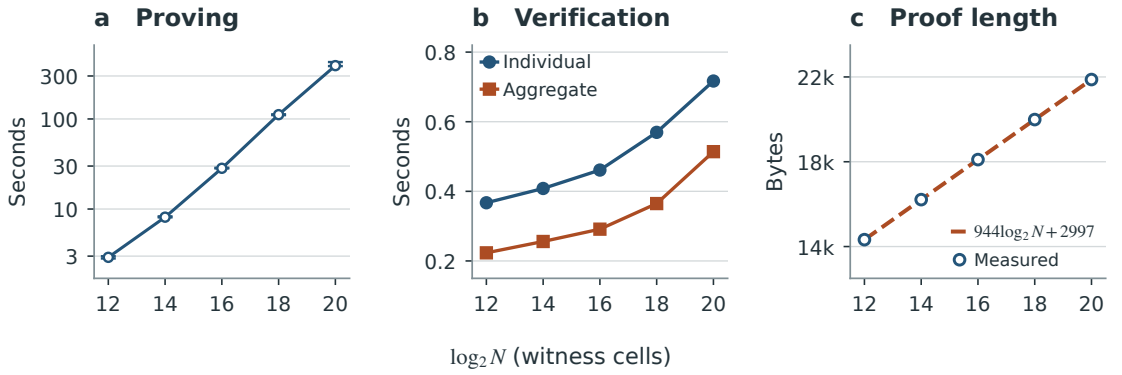

Fig. 5. Four-thread scale of tight-cap core7 on the Xeon E5-2696 v4. Prover bars give 95% intervals.

Median prover time rises from 2.909 to 390.834 seconds, and proof size from 14,325 to 21,877 bytes. At the largest point the prover-median interval is $[386.229, 428.394]$ seconds. Median individual and aggregate verification times are 0.717 and 0.514 seconds. The five measured lengths agree with $944 \log_2 N + 2997$ bytes. QG in the mechanism study uses $E = D$, which raises the final-round cap from seven to twelve and accounts for its additional $5 \cdot 32 = 160$ bytes.

At $N = 2^{20}$ the SRS occupies 251,668,264 bytes. Checked loading and cache reconstruction took 230.066 seconds, and index certification took 19.224 seconds. These are

single initialization observations with the operating system's page cache left in its existing state. They are excluded from online proving time. Appendix D.2 records the separate $N = 2^{22}$ observation obtained before that configuration was stopped.

7.4 Engineering tradeoffs

Quotient caching. For the thirteen-selector gate at $N = 1024$, $k = 8$, degree five, $\nu_p = 1$, and four threads, cached openings reduce median prover time from 1.722 to 1.320 seconds. The median within-witness direct/cached ratio is 1.330 $[1.254, 1.383]$ over ten matched inputs under blocked execution. Both paths produce 12,437-byte proofs.

Scalar lookup. For PowerCopy at the same size, width and degree, with $\nu_p = 1$, anchor three, $E = D$, and a two-value table containing zero, lookup raises the core7 length from 8,597 to 18,837 bytes and the core6 length from 9,653 to 19,893 bytes. Each increment is $3 \cdot 96 + 23 \cdot 32 + 16 \cdot 12 \cdot 48 = 10{,}240$ bytes for three source commitments, 23 endpoint scalars, and 16 buckets of twelve $\mathbb{G}_1$ elements. These extension costs are matched on parameters. Configuration-dependent lookup target generation changes the numerical witnesses.

Appendix D gives the parameter scans, thread and throughput measurements, and the comparison with B1, our same-backend trace construction. B1 changes masking, helper commitments, check organization, and support representation together. Its paired results measure the complete implementations. The three controls in Table 3 isolate the indicated protocol choices.

8 Conclusion

Zero knowledge in a polynomial argument is governed by the information available to the verifier. This information can include structured observations, linear combinations, and intermediate messages whose distributions depend on the preceding transcript. In this paper, we developed a conditional affine view of these observations and used it to construct low-degree masking spaces that match the information disclosed by a multilinear argument. The paired-tree construction illustrates the principle: nine random coefficients suffice to hide eight boundary evaluations and one root difference, attaining the dimension required by the specified uniform observation interface.

We realized this approach in GoZK, combining query-aware source masks with a causal sumcheck helper, global aggregation, and authenticated polynomial evaluations. The construction preserves the intended joint observation interface through the complete transcript and admits exact online simulation under the stated privacy conditions. Our implementation demonstrates that these choices have concrete consequences. Query-aware masking reduces source randomness, sumcheck communication, and prover time in the measured configurations, while global aggregation provides additional communication savings at a modest computational cost. The results also show that minimizing masking dimension or the number of checked relations alone is insufficient. Polynomial degrees,

commitment support, and the structure of authenticated openings must be considered together.

An interesting direction is to extend this approach from explicitly designed query interfaces to more general families of adaptive and correlated observations. In particular, the conditional affine characterization suggests the possibility of automatically deriving masking spaces from a protocol's observation structure, while jointly optimizing their dimension, degree, and commitment cost. Broader lookup relations, more complex cross-polynomial observations, and alternative commitment backends provide natural settings for such constructions. Extending the simulation framework to stronger security and composition models is another direction for future work.

The verifier's observable information is itself an algebraic object that can be analyzed and optimized. Using this object to choose a masking space connects the privacy requirements of a proof system to the cost of implementing it.

Disclosure of AI Assistance

The authors used AI tools to polish the prose, check the rigor of mathematical arguments, and suggest approaches to parts of the proofs within the framework the authors developed. AI tools also assisted in writing portions of the implementation. The authors reviewed all AI-assisted material and take responsibility for the paper and its accompanying code.

References

1. Abdolmaleki, B., Lipmaa, H., Siim, J., Zając, M.: On Subversion-Resistant SNARKs. Journal of Cryptology **34**(3) (2021). https://doi.org/10.1007/s00145-021-09379-y, https://eprint.iacr.org/2020/668, article 17
2. Albrecht, M.R., Fenzi, G., Lapiha, O., Nguyen, N.K.: SLAP: Succinct Lattice-Based Polynomial Commitments from Standard Assumptions. In: Advances in Cryptology – EUROCRYPT 2024. pp. 90–119 (2024). https://doi.org/10.1007/978-3-031-58754-2_4, https://eprint.iacr.org/2023/1469
3. Alon, N.: Combinatorial nullstellensatz. Combinatorics, Probability and Computing **8**(1–2), 7–29 (1999). https://doi.org/10.1017/S0963548398003411
4. Ames, S., Hazay, C., Ishai, Y., Venkitasubramaniam, M.: Ligero: Lightweight sublinear arguments without a trusted setup. In: ACM Conference on Computer and Communications Security (2017), https://acmccs.github.io/papers/p2087-amesA.pdf
5. Arun, A., Setty, S., Thaler, J.: Jolt: SNARKs for virtual machines via lookups. Cryptology ePrint Archive, Paper 2023/1217 (2023), https://eprint.iacr.org/2023/1217
6. Attema, T., Fehr, S., Klooß, M.: Fiat–Shamir Transformation of Multi-Round Interactive Proofs (Extended Version). Journal of Cryptology **36**(4) (2023). https://doi.org/10.1007/s00145-023-09478-y, https://ir.cwi.nl/pub/33324/, article 36
7. Attema, T., Fehr, S., Klooß, M., Resch, N.: The Fiat–Shamir Transformation of $(\Gamma_1, \ldots, \Gamma_\mu)$-Special-Sound Interactive Proofs. Journal of Cryptology **39**(2) (2026). https://doi.org/10.1007/s00145-025-09569-y, https://eprint.iacr.org/2023/1945, article 19
8. Bagad, S., Domb, Y., Thaler, J.: The sum-check protocol over fields of small characteristic. Cryptology ePrint Archive, Paper 2024/1046 (2024), https://eprint.iacr.org/2024/1046
9. Bauer, B., Fuchsbauer, G., Loss, J.: A Classification of Computational Assumptions in the Algebraic Group Model. In: Advances in Cryptology – CRYPTO 2020. pp. 121–151 (2020). https://doi.org/10.1007/978-3-030-56880-1_5, https://repositum.tuwien.at/handle/20.500.12708/58311

10. Baweja, A., Chiesa, A., Fedele, E., Fenzi, G., Mishra, P., Mopuri, T., Zitek-Estrada, A.: Time-space trade-offs for sumcheck. Cryptology ePrint Archive, Paper 2025/1473 (2025), https://eprint.iacr.org/2025/1473
11. Bellare, M., Rogaway, P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In: First ACM Conference on Computer and Communications Security. pp. 62–73 (1993). https://doi.org/10.1145/168588.168596, https://web.cs.ucdavis.edu/~rogaway/papers/ro-abstract.html
12. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Fast Reed–Solomon interactive oracle proofs of proximity. In: 45th International Colloquium on Automata, Languages, and Programming. pp. 14:1–14:17 (2018), https://doi.org/10.4230/LIPIcs.ICALP.2018.14
13. Ben-Sasson, E., Carmon, D., Ishai, Y., Kopparty, S., Saraf, S.: Proximity gaps for Reed–Solomon codes. Cryptology ePrint Archive, Paper 2020/654 (2020), https://eprint.iacr.org/2020/654
14. Ben-Sasson, E., Chiesa, A., Forbes, M.A., Gabizon, A., Riabzev, M., Spooner, N.: Zero knowledge protocols from succinct constraint detection. Tech. Rep. TR16-156, revision 1, Electronic Colloquium on Computational Complexity (2017), https://eccc.weizmann.ac.il/report/2016/156/, revision of 21 September 2017
15. Ben-Sasson, E., Chiesa, A., Gabizon, A., Riabzev, M., Spooner, N.: Interactive oracle proofs with constant rate and query complexity. In: 44th International Colloquium on Automata, Languages, and Programming. pp. 40:1–40:15 (2017), https://doi.org/10.4230/LIPIcs.ICALP.2017.40
16. Ben-Sasson, E., Chiesa, A., Green, M., Tromer, E., Virza, M.: Secure Sampling of Public Parameters for Succinct Zero Knowledge Proofs. In: 2015 IEEE Symposium on Security and Privacy. pp. 287–304 (2015). https://doi.org/10.1109/sp.2015.25, https://www.ieee-security.org/TC/SP2015/papers-archived/6949a287.pdf
17. Ben-Sasson, E., Chiesa, A., Riabzev, M., Spooner, N., Virza, M., Ward, N.P.: Aurora: Transparent succinct arguments for R1CS. In: Advances in Cryptology – EUROCRYPT (2019), https://doi.org/10.1007/978-3-030-17653-2_4
18. Ben-Sasson, E., Chiesa, A., Spooner, N.: Interactive oracle proofs. In: Theory of Cryptography – TCC. pp. 31–60 (2016), https://eprint.iacr.org/2016/116
19. Ben-Sasson, E., Sudan, M.: Short PCPs with polylog query complexity. SIAM Journal on Computing **38**(2), 551–607 (2008), https://doi.org/10.1137/050646445
20. Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In: Advances in Cryptology – ASIACRYPT 2011. pp. 41–69 (2011)
21. Bowe, S., Gabizon, A., Miers, I.: Scalable Multi-party Computation for zk-SNARK Parameters in the Random Beacon Model. Tech. Rep. 2017/1050, Cryptology ePrint Archive (2017), https://eprint.iacr.org/2017/1050
22. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: 2018 IEEE Symposium on Security and Privacy. pp. 315–334 (2018). https://doi.org/10.1109/SP.2018.00020
23. Bünz, B., Fisch, B., Szepieniec, A.: Transparent SNARKs from DARK Compilers. In: Advances in Cryptology – EUROCRYPT 2020. pp. 677–706 (2020). https://doi.org/10.1007/978-3-030-45721-1_24, https://eprint.iacr.org/2019/1229
24. Bush, K.A.: Orthogonal arrays of index unity. The Annals of Mathematical Statistics **23**(3), 426–434 (1952). https://doi.org/10.1214/aoms/1177729387, https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-23/issue-3/Orthogonal-Arrays-of-Index-Unity/10.1214/aoms/1177729387.full
25. Canetti, R., Chen, Y., Holmgren, J., Lombardi, A., Rothblum, G.N., Rothblum, R.D.: Fiat–Shamir From Simpler Assumptions. Tech. Rep. 2018/1004, Cryptology ePrint Archive (2018), https://eprint.iacr.org/2018/1004
26. Chen, B., Bünz, B., Boneh, D., Zhang, Z.: HyperPlonk: Plonk with linear-time prover and high-degree custom gates. In: Advances in Cryptology – EUROCRYPT 2023, Part II. pp. 499–530. Springer (2023). https://doi.org/10.1007/978-3-031-30617-4_17, https://eprint.iacr.org/2022/1355, full version: IACR ePrint 2022/1355
27. Chiesa, A., Forbes, M.A., Spooner, N.: A zero knowledge sumcheck and its applications. Cryptology ePrint Archive, Paper 2017/305 (2017), https://eprint.iacr.org/2017/305

28. Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, P., Ward, N.: Marlin: Preprocessing zkSNARKs with universal and updatable SRS. Cryptology ePrint Archive, Paper 2019/1047 (2019), https://eprint.iacr.org/2019/1047
29. Cormode, G., Mitzenmacher, M., Thaler, J.: Practical verified computation with streaming interactive proofs. In: Innovations in Theoretical Computer Science (2012), https://arxiv.org/abs/1105.2003
30. Cormode, G., Thaler, J., Yi, K.: Verifying computations with streaming interactive proofs. Proceedings of the VLDB Endowment **5**(1) (2011), https://arxiv.org/abs/1109.6882
31. Daftardar, A., Mo, J., Ah-kiow, J., Bünz, B., Karri, R., Garg, S., Reagen, B.: Need for zkSpeed: Accelerating HyperPlonk for zero-knowledge proofs. Cryptology ePrint Archive, Paper 2025/620 (2025). https://doi.org/10.1145/3695053.3731021, https://eprint.iacr.org/2025/620
32. Dao, Q., Miller, J., Wright, O., Grubbs, P.: Weak Fiat–Shamir Attacks on Modern Proof Systems. In: 2023 IEEE Symposium on Security and Privacy (SP). pp. 199–216 (2023). https://doi.org/10.1109/SP46215.2023.10179408, https://eprint.iacr.org/2023/691
33. DeMillo, R.A., Lipton, R.J.: A probabilistic remark on algebraic program testing. Information Processing Letters **7**(4), 193–195 (1978). https://doi.org/10.1016/0020-0190(78)90067-4, https://www.sciencedirect.com/science/article/pii/0020019078900674
34. Don, J., Fehr, S., Majenz, C.: The Measure-and-Reprogram Technique 2.0: Multi-Round Fiat–Shamir and More. In: Advances in Cryptology – CRYPTO 2020. pp. 602–631 (2020). https://doi.org/10.1007/978-3-030-56877-1_21, https://eprint.iacr.org/2020/282
35. Don, J., Fehr, S., Majenz, C., Schaffner, C.: Security of the Fiat–Shamir Transformation in the Quantum Random-Oracle Model. In: Advances in Cryptology – CRYPTO 2019. pp. 356–383 (2019). https://doi.org/10.1007/978-3-030-26951-7_13, https://eprint.iacr.org/2019/190
36. Don, J., Fehr, S., Majenz, C., Schaffner, C.: Online-Extractability in the Quantum Random-Oracle Model. In: Advances in Cryptology – EUROCRYPT 2022. pp. 677–706 (2022). https://doi.org/10.1007/978-3-031-07082-2_24, https://arxiv.org/abs/2103.03085
37. Eagen, L., Fiore, D., Gabizon, A.: cq: Cached quotients for fast lookups. Cryptology ePrint Archive, Paper 2022/1763 (2022), https://eprint.iacr.org/2022/1763, revision of 5 July 2024
38. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Advances in Cryptology – CRYPTO '86. pp. 186–194 (1987)
39. Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: Advances in Cryptology – CRYPTO 2018, Part II. pp. 33–62. Springer (2018). https://doi.org/10.1007/978-3-319-96881-0_2, https://eprint.iacr.org/2017/620
40. Gabizon, A., Williamson, Z.J.: plookup: A simplified polynomial protocol for lookup tables. Cryptology ePrint Archive, Paper 2020/315 (2020), https://eprint.iacr.org/2020/315
41. Gabizon, A., Williamson, Z.J., Ciobotaru, O.: PLONK: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Paper 2019/953 (2019), https://eprint.iacr.org/2019/953
42. Goldwasser, S., Kalai, Y.T., Rothblum, G.N.: Delegating computation: Interactive proofs for muggles. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing. pp. 113–122 (2008)
43. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM Journal on Computing **18**(1), 186–208 (1989). https://doi.org/10.1137/0218012
44. Golovnev, A., Lee, J., Setty, S., Thaler, J., Wahby, R.S.: Brakedown: Linear-time and field-agnostic SNARKs for R1CS. In: Advances in Cryptology – CRYPTO (2023), https://doi.org/10.1007/978-3-031-38545-2_7
45. Gorbunov, S., Reyzin, L., Wee, H., Zhang, Z.: Pointproofs: Aggregating Proofs for Multiple Vector Commitments. In: ACM Conference on Computer and Communications Security. pp. 2007–2023 (2020). https://doi.org/10.1145/3372297.3417244, https://eprint.iacr.org/2020/419
46. Groth, J.: On the size of pairing-based non-interactive arguments. Cryptology ePrint Archive, Paper 2016/260 (2016), https://eprint.iacr.org/2016/260
47. Gur, T., O'Connor, J., Spooner, N.: Perfect zero-knowledge PCPs for #p. In: Proceedings of the 56th Annual ACM Symposium on Theory of Computing (2024), https://doi.org/10.1145/3618260.3649698
48. Gur, T., O'Connor, J., Spooner, N.: A zero-knowledge PCP theorem. In: Proceedings of the 57th Annual ACM Symposium on Theory of Computing (2025), https://doi.org/10.1145/3717823.3718128

49. Haböck, U.: Multivariate lookups based on logarithmic derivatives. Cryptology ePrint Archive, Paper 2022/1530 (2022), https://eprint.iacr.org/2022/1530
50. Hamming, R.W.: Error detecting and error correcting codes. Bell System Technical Journal **29**(2), 147–160 (1950). https://doi.org/10.1002/j.1538-7305.1950.tb00463.x, https://onlinelibrary.wiley.com/doi/10.1002/j.1538-7305.1950.tb00463.x
51. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge proofs from secure multiparty computation. SIAM Journal on Computing **39**(3), 1121–1152 (2009), https://doi.org/10.1137/080725398
52. Ishai, Y., Mahmoody, M., Sahai, A.: On efficient zero-knowledge PCPs. In: Theory of Cryptography – TCC. pp. 151–168 (2012), https://doi.org/10.1007/978-3-642-28914-9_9
53. Kate, A., Zaverucha, G.M., Goldberg, I.: Constant-size commitments to polynomials and their applications. In: Advances in Cryptology – ASIACRYPT 2010. pp. 177–194 (2010)
54. Kilian, J., Petrank, E., Tardos, G.: Probabilistically checkable proofs with zero knowledge. In: Proceedings of the 29th Annual ACM Symposium on Theory of Computing. pp. 496–505 (1997), https://doi.org/10.1145/258533.258643
55. Kothapalli, A., Setty, S.: HyperNova: Recursive arguments for customizable constraint systems. Cryptology ePrint Archive, Paper 2023/573 (2023), https://eprint.iacr.org/2023/573
56. Kothapalli, A., Setty, S., Tzialla, I.: Nova: Recursive zero-knowledge arguments from folding schemes. Cryptology ePrint Archive, Paper 2021/370 (2021), https://eprint.iacr.org/2021/370
57. Lee, J.: Dory: Efficient, Transparent arguments for Generalised Inner Products and Polynomial Commitments. In: Theory of Cryptography – TCC 2021. pp. 1–34 (2021). https://doi.org/10.1007/978-3-030-90453-1_1, https://eprint.iacr.org/2020/1274
58. Li, C., Zhu, P., Li, Y., Hong, C., Qu, W., Zhang, J.: HyperPianist: Pianist with linear-time prover and logarithmic communication cost. Cryptology ePrint Archive, Paper 2024/1273 (2024), https://eprint.iacr.org/2024/1273
59. Libert, B.: Simulation-extractable KZG polynomial commitments and applications to HyperPlonk. In: Public-Key Cryptography – PKC 2024, Part II. pp. 68–98. Springer (2024), https://eprint.iacr.org/2024/854, full version: IACR ePrint 2024/854
60. Liu, Q., Zhandry, M.: Revisiting Post-Quantum Fiat–Shamir. In: Advances in Cryptology – CRYPTO 2019. pp. 326–355 (2019). https://doi.org/10.1007/978-3-030-26951-7_12, https://eprint.iacr.org/2019/262
61. Liu, T., Xie, T., Zhang, J., Song, D., Zhang, Y.: Pianist: Scalable zkRollups via fully distributed zero-knowledge proofs. Cryptology ePrint Archive, Paper 2023/1271 (2023), https://eprint.iacr.org/2023/1271
62. Lund, C., Fortnow, L., Karloff, H., Nisan, N.: Algebraic methods for interactive proof systems. Journal of the ACM **39**(4), 859–868 (1992). https://doi.org/10.1145/146585.146605
63. Maller, M., Bowe, S., Kohlweiss, M., Meiklejohn, S.: Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. pp. 2111–2128 (2019). https://doi.org/10.1145/3319535.3339817, https://doi.org/10.1145/3319535.3339817
64. Nguyen, N.K., Seiler, G.: Greyhound: Fast Polynomial Commitments from Lattices. In: Advances in Cryptology – CRYPTO 2024. pp. 243–275 (2024). https://doi.org/10.1007/978-3-031-68403-6_8, https://eprint.iacr.org/2024/1293
65. Ozarow, L.H., Wyner, A.D.: Wire-tap channel II. In: Beth, T., Cot, N., Ingemarsson, I. (eds.) Advances in Cryptology: Proceedings of EUROCRYPT 84. Lecture Notes in Computer Science, vol. 209, pp. 33–50. Springer (1985). https://doi.org/10.1007/3-540-39757-4_5, https://link.springer.com/chapter/10.1007/3-540-39757-4_5
66. Parno, B., Gentry, C., Howell, J., Raykova, M.: Pinocchio: Nearly practical verifiable computation. Cryptology ePrint Archive, Paper 2013/279 (2013), https://eprint.iacr.org/2013/279
67. Pearson, L., Fitzgerald, J., Masip, H., Bellés-Muñoz, M., Muñoz-Tapia, J.L.: PlonKup: Reconciling PlonK with plookup. Cryptology ePrint Archive, Paper 2022/086 (2022), https://eprint.iacr.org/2022/086
68. Pointcheval, D., Stern, J.: Security Arguments for Digital Signatures and Blind Signatures. Journal of Cryptology **13**(3), 361–396 (2000). https://doi.org/10.1007/s001450010003, https://www.di.ens.fr/david.pointcheval/Documents/Papers/2000_joc.pdf

69. Posen, J., Kattis, A.A.: Caulk+: Table-independent lookup arguments. Cryptology ePrint Archive, Paper 2022/957 (2022), https://eprint.iacr.org/2022/957
70. Rao, C.R.: Factorial experiments derivable from combinatorial arrangements of arrays. Supplement to the Journal of the Royal Statistical Society **9**(1), 128–139 (1947), https://academic.oup.com/jrsssb/article/9/1/128/7026526
71. Rotem, L., Tessaro, S.: Straight-Line Knowledge Extraction for Multi-Round Protocols. In: Advances in Cryptology – CRYPTO 2025. pp. 95–127 (2025). https://doi.org/10.1007/978-3-032-01907-3_4, https://eprint.iacr.org/2024/1724
72. Schwartz, J.T.: Fast probabilistic algorithms for verification of polynomial identities. Journal of the ACM **27**(4), 701–717 (1980). https://doi.org/10.1145/322217.322225
73. Sefranek, M.: How (not) to simulate PLONK. In: Security and Cryptography for Networks – SCN 2024, Part I. pp. 96–117. Springer (2024), https://eprint.iacr.org/2024/848, full version: IACR ePrint 2024/848
74. Setty, S.: Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In: Advances in Cryptology – CRYPTO 2020, Part III. pp. 704–737 (2020). https://doi.org/10.1007/978-3-030-56877-1_25
75. Setty, S., Thaler, J., Wahby, R.: Customizable constraint systems for succinct arguments. Cryptology ePrint Archive, Paper 2023/552 (2023), https://eprint.iacr.org/2023/552
76. Setty, S., Thaler, J., Wahby, R.: Unlocking the lookup singularity with lasso. Cryptology ePrint Archive, Paper 2023/1216 (2023), https://eprint.iacr.org/2023/1216
77. Setty, S., Thaler, J., Zhu, M.: Twist and shout: Faster memory checking arguments via one-hot addressing and increments. Cryptology ePrint Archive, Paper 2025/105 (2025), https://eprint.iacr.org/2025/105
78. Shamir, A.: How to share a secret. Communications of the ACM **22**(11), 612–613 (1979). https://doi.org/10.1145/359168.359176, https://dl.acm.org/doi/10.1145/359168.359176
79. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. arXiv:quant-ph/9508027 (1996), https://arxiv.org/abs/quant-ph/9508027, revised version of 25 January 1996
80. Shoup, V.: Lower Bounds for Discrete Logarithms and Related Problems. In: Advances in Cryptology – EUROCRYPT '97. pp. 256–266 (1997). https://doi.org/10.1007/3-540-69053-0_18, https://doi.org/10.1007/3-540-69053-0_18
81. Thaler, J.: Time-optimal interactive proofs for circuit evaluation. In: Advances in Cryptology – CRYPTO. pp. 71–89 (2013), https://people.cs.georgetown.edu/jthaler/TimeOptimalIPs.html
82. Thaler, J.: Proofs, arguments, and zero-knowledge (2023), https://people.cs.georgetown.edu/jthaler/ProofsArgsAndZK.pdf, author's manuscript, 18 July 2023
83. Unruh, D.: Post-Quantum Security of Fiat–Shamir. In: Advances in Cryptology – ASIACRYPT 2017. pp. 65–95 (2017). https://doi.org/10.1007/978-3-319-70694-8_3, https://eprint.iacr.org/2017/398
84. Vu, V., Setty, S., Blumberg, A.J., Walfish, M.: A hybrid architecture for interactive verifiable computation. In: IEEE Symposium on Security and Privacy (2013), https://www.cs.utexas.edu/lasr/paper.php?uid=58
85. Wahby, R.S., Tzialla, I., abhi shelat, Thaler, J., Walfish, M.: Doubly-efficient zkSNARKs without trusted setup. In: IEEE Symposium on Security and Privacy (2018), https://eprint.iacr.org/2017/1132
86. Wu, H., Zheng, W., Chiesa, A., Popa, R.A., Stoica, I.: DIZK: A distributed zero knowledge proof system. Cryptology ePrint Archive, Paper 2018/691 (2018), https://eprint.iacr.org/2018/691
87. Xie, T., Zhang, J., Cheng, Z., Zhang, F., Zhang, Y., Jia, Y., Boneh, D., Song, D.: zkbridge: Trustless cross-chain bridges made practical. arXiv:2210.00264 (2022), https://arxiv.org/abs/2210.00264
88. Xie, T., Zhang, J., Zhang, Y., Papamanthou, C., Song, D.: Libra: Succinct zero-knowledge proofs with optimal prover computation. Cryptology ePrint Archive, Paper 2019/317 (2019), https://eprint.iacr.org/2019/317, full version of the CRYPTO 2019 paper
89. Xie, T., Zhang, Y., Song, D.: Orion: Zero knowledge proof with linear prover time. In: Advances in Cryptology – CRYPTO (2022), https://doi.org/10.1007/978-3-031-15985-5_11

90. Zapico, A., Buterin, V., Khovratovich, D., Maller, M., Nitulescu, A., Simkin, M.: Caulk: Lookup arguments in sublinear time. Cryptology ePrint Archive, Paper 2022/621 (2022), https://eprint.iacr.org/2022/621
91. Zapico, A., Gabizon, A., Khovratovich, D., Maller, M., Ràfols, C.: Baloo: Algebraic lookup arguments. IACR Communications in Cryptology **2**(2) (2025). https://doi.org/10.62056/ae890l5vt, https://doi.org/10.62056/ae890l5vt
92. Zhang, J., Xie, T., Zhang, Y., Song, D.: Transparent polynomial delegation and its applications to zero knowledge proof. In: IEEE Symposium on Security and Privacy (2020), https://eprint.iacr.org/2019/1482
93. Zippel, R.: Probabilistic algorithms for sparse polynomials. In: Ng, E.W. (ed.) Symbolic and Algebraic Computation: EUROSAM '79. Lecture Notes in Computer Science, vol. 72, pp. 216–226. Springer (1979). https://doi.org/10.1007/3-540-09519-5_73, https://link.springer.com/chapter/10.1007/3-540-09519-5_73

Appendix A Proofs for Query-Aware Masking

A.1 Affine fibers under adaptive observations

For a fixed candidate history h, write the observed constraints on the initial uniform mask as $A_h R = z_h - b_h(s)$. Every feasible mask gives the same public responses along this history. A public adaptive strategy assigns the same probability to its transitions for every such mask. Conditioning on h therefore leaves the mask uniform on the fiber. Lemma 3.1 gives the next conditional law. If the two histories have the same distribution so far, equal response supports at every reachable history and common public transitions extend that equality by one message. Section 5.2 establishes the corresponding posterior for the complete proof transcript.

Private differences. For a fixed observation map E, the law of $E(p+R)$ for $R \leftarrow \mathrm{Unif}(\mathcal{A})$ is uniform on $E(p) + E(\mathcal{A})$. Write $\mathcal{P}$ for the span of all admissible differences $p - p'$. The supports agree exactly under the condition

$$E(\mathcal{P}) \subseteq E(\mathcal{A}). \tag{26}$$

If p itself is uniform on $\mathcal{P}$ and independent of R, the mutual information is $(\dim(E\mathcal{P} + E\mathcal{A}) - \dim E\mathcal{A}) \log_2 q$. The two dimensions give the output entropy and its conditional entropy in units of $\log_2 q$. In Shamir's threshold scheme, uniform higher coefficients make any set of fewer than the threshold number of shares independent of the secret constant term [78]. Ozarow and Wyner use random coset encoding to hide a message from an observer who chooses a subset of the transmitted coordinates [65].

Joint observations. Fix a basis $f_1, \ldots, f_d$ of $\mathcal{A}$. The answers on a set S of complete points are jointly uniform precisely when the columns $(f_1(x), \ldots, f_d(x))^{\mathsf{T}}$ for $x \in S$ are linearly independent. In a downward-closed allowed family, adjoining a new independent column leaves the new answer uniform conditional on previous answers. A repeated column returns the previous answer. The same argument applies to linear functionals of several sources by using their joint observation matrix.

A.2 Interpolation on Cartesian grids

Choose finite nonempty proper sets $A_i \subset \mathbb{F}$, put $a_i = |A_i|$, and define $\phi_i(X_i) = \prod_{u \in A_i}(X_i - u)$. Write $\ell_{i,u}$ for the Lagrange basis on A_i. Let $\Sigma \subseteq \prod_i (A_i \cup \{*\})$ be any collection of states, each with at least one star. For $\sigma \in \Sigma$ set

$$b_\sigma(X) = \prod_{\sigma_i = *} \phi_i(X_i) \prod_{\sigma_i \in A_i} \ell_{i,\sigma_i}(X_i).$$

Each fixed coordinate of x^σ has its specified grid value and each starred coordinate lies outside its grid.

Lemma A.1 (Cartesian interpolation). *Evaluation on one representative x^σ per selected state maps* $\operatorname{span}\{b_\sigma : \sigma \in \Sigma\}$ *bijectively to $\mathbb{F}^\Sigma$. Every basis polynomial vanishes on $\prod_i A_i$ and has degree at most a_i in coordinate i.*

Proof. Write $S(\sigma) = \{i : \sigma_i = *\}$. A nonzero matrix entry $b_\tau(x^\sigma)$ requires $S(\tau) \subseteq S(\sigma)$. With equal star sets, distinct states differ at a grid coordinate where the Lagrange factor vanishes. Ordering by star-set size thus gives a triangular matrix with diagonal $\prod_{i \in S(\sigma)} \phi_i(x_i^\sigma) \neq 0$. The vanishing and degree bounds follow directly from the displayed product.

For the complete family of non-grid states, the univariate polynomials $\{\ell_{i,u} : u \in A_i\} \cup \{\phi_i\}$ form a basis of the polynomials of degree at most a_i. Their tensor products without a ϕ_i restrict to a basis of all grid functions. Those containing at least one ϕ_i form precisely the vanishing space inside the degree box. This is the individual-degree part of the Cartesian grid vanishing ideal [3, Theorem 1.1]. Its dimension is

$$\prod_i (a_i + 1) - \prod_i a_i. \tag{27}$$

Proposition A.2 (Coordinate-degree necessity). *If an allowed state has a single star in coordinate i, any mask space vanishing on $\prod_j A_j$ that produces a uniform answer there contains a mask of coordinate degree at least a_i.*

Proof. Fix the other coordinates to their grid values. Every restricted mask vanishes on all a_i values of A_i. A univariate polynomial of degree less than a_i with these roots is zero everywhere, including at the requested point.

A.3 Root combinations for several trees

A single tree uses the four boundary directions of Construction 1 and the fifth direction $\psi(u, v) = \phi(u)\phi(v)$. The four boundary values determine the first four coefficients, and the fifth value determines the remaining coefficient because ψ is nonzero there. For r trees, suppose the required root observations are the rows of a fixed rank-r_0 matrix M applied to their terminal values. Each tree also supplies four boundary values.

Proposition A.3 (Joint dimension for several trees). *The full uniform interface of $4r$ boundary values and r_0 independent root combinations admits a mask of dimension $4r + r_0$. This dimension is minimal for that interface.*

Proof. Choose r_0 independent rows and a column basis J of M before sampling the masks. Give every tree four independent boundary coefficients. Give only trees in J an extra ψ coefficient. After fixing all boundary values, the remaining matrix on the chosen rows is $M_J \operatorname{diag}(\psi_j(r_j) : j \in J)$. Both factors are invertible. The other root combinations follow by linearity. Surjectivity onto $\mathbb{F}^{4r+r_0}$ requires at least $4r + r_0$ dimensions. A family of later choices of M may use the same construction when the preselected columns span the root image for every allowed choice.

A.4 Witness, sequence, and degenerate profiles

Column and sequence pivots. For the complete lookup witness interface the mask is $\phi(Z)D(C) + \phi(X_\nu)D'(C) + \phi(C_j)(d_0 + d_1 Z)$. At a lookup point the first and third terms vanish. Dividing by $\phi(X_\nu)$ recovers $D'(a)$ for each Boolean address a. The resulting multilinear polynomial is known. Subtract its contribution at the gate points and divide by $\phi(Z)$ to obtain D on all Boolean columns. The copy point determines d_0, and the native public-input point determines d_1 with pivot $Z\phi(C_j)$. In the core the D' term is absent. With a virtual public slice the public-input point and d_1 disappear. The resulting witness dimensions are $k + 1$ for core6 and $2k + 1$ for lookup11.

At the sequence queries $(0, *), (1, *), (*, *)$, the mask is $a\phi(v) + bu\phi(v) + c\phi(u)$. The first two evaluations determine a, b in order. The last determines c. A complemented non-Boolean coordinate remains non-Boolean. The public factor $1 - \operatorname{eq}(Y, 0)$ then fixes the semantic sequence value at zero without requiring an opening of the underlying source there.

Boundary dimensions. For $c = 0, \nu \geq 1$, suffix alignment identifies the copy and lookup witness points, when present, at $(0, r_{L-\nu+1}, \ldots, r_L)$. It identifies the gate and native public-input points at $(r_{L-\nu}, \ldots, r_L)$. Identical prefix padding gives complete-point equalities. The first observation determines b in $a\phi(Z) + b\phi(X_\nu)$ and the second determines a.

At $N = 1$ the one remaining witness coordinate uses $a\phi(Z)$ and there is no copy tree. For $\nu = 0$ and $c \geq 1$ the core still has distinct column addresses and a tree of dimension $c + 1 \geq 2$. The last tree coordinate is now a column coordinate. Shared directions must be counted once when forming the actual support. Lookup requires $\nu \geq 1$.

A.5 The causal helper and its entropy bound

Proof (Proof of Proposition 4.1). Write

$$A(T) = \sum_{e=2}^{E_1} a_e(T^e - T), \qquad C_a(T) = \sum_{e=1}^{E_a} c_{a,e}(T^e - \tfrac{1}{2}).$$

The first-round helper polynomial is $2^{L-1}A(T)$, because every C_a has Boolean sum zero. For $e \geq 2$, the transmitted coefficient equals $f_{1,e} + \rho 2^{L-1} a_e$. These coefficients are uniform and independent for each fixed source state; the two remaining coefficients are exactly (10). At round $a \geq 2$, the helper partial sum is

$$2^{L-a}\left(A(r_1) + \sum_{b=2}^{a-1} C_b(r_b) + C_a(T)\right). \tag{28}$$

Consequently, each positive coefficient is $f_{a,e} + \rho 2^{L-a} c_{a,e}$. No future coefficient block occurs in the complete previous messages, including their constant terms. The current block is therefore fresh after conditioning on the preceding history. Odd characteristic and $\rho \neq 0$ make every scale factor invertible. The endpoint-sum condition determines the constant by (11).

A verifier may choose the next legal challenge after receiving the current message. For each fixed source state and each legal complete round transcript, the displayed equations uniquely recover the helper coefficients. Hence every source state assigns the same probability to that transcript. The helper round messages introduce no additional constraint on the source posterior. If a helper commitment is present, this last statement requires its blinding distribution to preserve the stated posterior, as established by the protocol's commitment kernel.

Proposition A.4 (Rank and entropy for the complete Boolean-zero family). *Let $\mathcal{I}_E$ consist of all polynomials with $\deg_{X_a} F \leq E_a$ and $F|_{\{0,1\}^L} = 0$. Assume $E_1 \geq 2$, $E_a \geq 1$ for $a > 1$, and $r_1 \notin \{0,1\}$. On a fixed path r, the rank of the free round-coefficient observation of $\mathcal{I}_E$ is $t-1$, where $t = \sum_a E_a$. Every independent additive noise distribution that hides all these differences has entropy at least $(t-1)\log_2 q$. The helper in (8) attains the bound.*

Proof. The first-round endpoints are zero, leaving $E_1 - 1$ free high coefficients. Each later round has E_a free positive coefficients; its constant is determined by the current claim. Thus the observation rank is at most $t-1$. The following polynomials lie in $\mathcal{I}_E$:

$$\begin{aligned} &X_1^e - X_1 && (2 \leq e \leq E_1),\\ &X_1(X_1-1)(X_a^e - \tfrac{1}{2}) && (2 \leq a \leq L,\ 1 \leq e \leq E_a). \end{aligned}$$

The first family isolates the first-round high coefficients, with nonzero scale 2^{L-1}, and contributes no later positive coefficients. In the second family, every earlier round is zero because summation over X_a cancels. At round a its unique positive coefficient is in position e and equals $r_1(r_1-1)2^{L-a} \neq 0$. Later positive coefficients are zero. These $t-1$ images are independent, proving equality of the rank.

If independent additive noise Z hides a linear translation space D, its law satisfies $Z \overset{d}{=} Z + d$ for every $d \in D$. It is consequently constant on each coset of D, and conditioned on any positive-mass coset it is uniform there. Thus $H(Z) \geq \log_2 |D|$. Taking $\dim D = t-1$ proves the entropy bound. The helper uses exactly $t-1$ uniform coefficients and maps bijectively to the free observations.

For the larger family with only the two half-cube sums fixed, the same rank is attained directly by the helper polynomials, for any fixed challenge path. If only the total sum is fixed, the zero-total-sum helper

$$g_0(X) = \sum_{a=1}^{L} \sum_{e=1}^{E_a} c_{a,e}(X_a^e - \tfrac{1}{2})$$

gives t independent round coordinates, and the complete total-sum family requires that many dimensions. A uniform linear helper whose total-sum functional is nonzero and whose sum is publicly sent needs at least $t+1$ dimensions. Its sum consumes one dimension, and each sum fiber must retain t dimensions for the rounds. At individual degree one the Boolean-zero family is $\{0\}$. For a particular circuit the lower bound depends on its own residual-difference space.

A.6 The stronger arbitrary-point requirement

Suppose the interface allows every set of at most T distinct non-Boolean points. If every T columns of the d-row evaluation matrix are independent, the table of answers over all q^d coefficient choices is an orthogonal array of strength T and index q^{d-T}. Bush studied the index-one case [24]. The proof below applies the sphere-packing count of Hamming [50] over $\mathbb{F}_q$ after quotienting by a chosen set of columns.

Proposition A.5 (Shortened syndrome bound). *Let a $d \times M$ matrix over $\mathbb{F}_q$ have every set of at most T columns linearly independent, with $T \le M$. For every $0 \le s \le T$,*

$$q^{d-s} \ge \sum_{j=0}^{\lfloor (T-s)/2 \rfloor} \binom{M-s}{j} (q-1)^j. \tag{29}$$

For evaluations on all non-Boolean points in L variables, $M = q^L - 2^L$. If $T = 2t+1$, then, for fixed L, t and sufficiently large odd primes q, this implies $d \ge 1 + t(L+1)$.

Proof. Choose s columns and quotient their span. It has dimension s; the remaining columns lie in a space of dimension $d-s$ and any at most $T-s$ are independent. If two error vectors of weight at most $\lfloor (T-s)/2 \rfloor$ had the same image under this quotient matrix, their difference would give a forbidden dependence on at most $T-s$ remaining columns. Their syndromes are distinct, proving (29). For odd T, take $s = 1$ and multiply by q. The right-hand side has leading order $q^{1+t(L+1)}/t!$ when $t > 0$; integrality of d gives the stated bound for sufficiently large q. For $t = 0$, one nonzero evaluation column already requires $d \ge 1$.

The odd-strength expression exceeds the unshortened radius-t expression by exactly $\binom{M-1}{t}(q-1)^{t+1}$. This is the linear syndrome proof of the odd-strength Rao bound [70]. It imposes no such bound on a restricted family that omits most T-point sets.

A.7 An explicit entropy–degree tradeoff

Let $\mathbb{F} = \mathbb{F}_q$ for an odd prime q, let $\mathbb{K} = \mathbb{F}_{q^L}$, and fix an $\mathbb{F}$-basis $\beta_1, \dots, \beta_L$. Define

$$\mathsf{L}(X) = \sum_i \beta_i X_i, \qquad \mathsf{H}(X) = \sum_i \beta_i \phi(X_i), \qquad \mu(X) = \prod_{j=0}^{L-1} \mathsf{H}^{[q^j]}(X).$$

The superscript $[q^j]$ applies Frobenius to coefficients only; variable exponents do not change. Frobenius permutes the factors, so μ has coefficients in $\mathbb{F}$ and total degree $2L$. At base-field points it equals the extension norm of $\mathsf{H}(x)$. The basis property gives $\mathsf{H}(x) = 0$ exactly when every $x_i \in \{0, 1\}$. Thus the base-field zero set of μ is exactly the Boolean cube.

For $1 \le T \le q - 2$, consider the formal polynomial space

$$\mathcal{A}_{\mathrm{norm}} = \mu(X)\, \mathrm{span}_{\mathbb{F}} \left(\{1\} \cup \{ \mathrm{Tr}^{\mathrm{coeff}}(a\mathsf{L}(X)^j) : a \in \mathbb{K},\ 1 \le j < T \} \right). \tag{30}$$

Here $\mathrm{Tr}^{\mathrm{coeff}}$ applies the field trace coefficientwise. This space has dimension $1 + L(T - 1)$ and total degree at most $2L + T - 1$; the same number bounds its individual degrees. It has full evaluation rank on every set of at most T distinct non-Boolean base-field points.

Evaluation rank. Suppose coefficients $\lambda_i \in \mathbb{F}$ annihilate all evaluations on such a set $\{x_i\}_{i=1}^u$, $u \le T$, and put $w_i = \lambda_i \mu(x_i)$. The constant direction gives $\sum_i w_i = 0$. Nondegeneracy of the trace pairing gives

$$\sum_i w_i \mathsf{L}(x_i)^j = 0 \quad (1 \le j < T).$$

The map $\mathsf{L} : \mathbb{F}^L \to \mathbb{K}$ is injective. The first u equations form an invertible Vandermonde system, so all w_i, and hence all λ_i, vanish.

Dimension and degree. The positive-j trace subspaces are homogeneous of different degrees. Each has dimension L. Indeed, if $\mathrm{Tr}^{\mathrm{coeff}}(a\mathsf{L}^j) = 0$, fix i and examine the coefficients of X_i^j and $X_i^{j-1} X_h$ for $h \ne i$. Since $j < q$, these imply $\mathrm{Tr}(a\beta_i^{j-1}\beta_h) = 0$ for every h. The trace pairing is nondegenerate, and multiplication by $\beta_i^{j-1} \ne 0$ preserves a basis, so $a = 0$. For $j = 1$, the coefficients of the X_h give the same conclusion directly. The constant direction contributes one dimension. Multiplication by the nonzero formal polynomial μ preserves linear independence. The degree bound follows from the displayed factors.

For the eight-column comparison, apply this same construction to the projected spaces $(L, T) = (4, 10)$ and $(2, 5)$. It uses 37 coefficients for the witness and 9 per tree. Its total-degree bounds are 17 and 8. Section A.8 specifies the different polynomial supports used in the experiment.

A general restriction follows from the common zero set. If d polynomials of total degree at most Δ have common base-field zero set exactly $\{0, 1\}^L$, then

$$d\Delta \ge L. \tag{31}$$

Otherwise $\prod_{i=1}^{d}(1-f_i^{q-1})$ has total degree smaller than $L(q-1)$. Summing any of its monomials over $\mathbb{F}_q^L$ gives zero, since some coordinate exponent is smaller than $q-1$. Its sum must therefore be zero in $\mathbb{F}_q$. But it is also the indicator of the common zero set, whose sum is $2^L \neq 0$ in odd characteristic, a contradiction. Restricted-query masks can have additional common zeros outside their permitted query family, so this obstruction does not apply to them in the same way.

A.8 A generic mask on projected queries

The generic experimental arm fixes a projection dimension $s \geq 1$ and a budget $T \geq 1$. Its mask handles every set of at most T distinct projected points outside the Boolean cube. Define

$$\Lambda_T = \left\{ \alpha \in \mathbb{Z}_{\geq 0}^s : \prod_{i=1}^{s}(1+\alpha_i) \leq T \right\}, \qquad J_T = {\downarrow} \bigcup_{i=1}^{s} (\Lambda_T + 2e_i).$$

The arrow denotes closure under coordinatewise decrease and e_i is a unit vector. Set $b(\alpha)_i = \mathbf{1}_{\alpha_i > 0}$. Use an independent uniform coefficient for each polynomial in

$$\mathcal{B}_{s,T} = \{X^\alpha - X^{b(\alpha)} : \alpha \in J_T,\ \max_i \alpha_i \geq 2\}. \tag{32}$$

Proposition A.6. *For every $t \leq T$ and every set of t distinct points in $\mathbb{F}^s \setminus \{0,1\}^s$, evaluation of the span of (32) surjects onto $\mathbb{F}^t$. Its mask preserves the entire Boolean table.*

Proof. Order the monomials by a graded monomial order. Keep a monomial if its evaluation vector at the t points is independent of the previously kept vectors. These vectors span $\mathbb{F}^t$: a product of separating linear factors interpolates each coordinate vector. The kept monomials form a lower set. Indeed, a discarded vector is a linear combination of earlier ones, and multiplying that relation by a monomial gives a relation in the same order for each of its multiples. There are exactly t kept monomials. Every divisor of a kept X^α also belongs to that set, which gives $\prod_i(1+\alpha_i) \leq t$. Thus Λ_T suffices to interpolate on these points.

For a target point x, let p_x be its cardinal interpolant with support in Λ_T. Choose j with $x_j \notin \{0,1\}$. Multiplying p_x by $\phi(X_j)/\phi(x_j)$ preserves its cardinal values and makes it zero on the cube. Its support lies in J_T. Boolean reduction sends X^α to $X^{b(\alpha)}$ and gives the unique multilinear representative on the cube. The new polynomial has zero Boolean reduction, so it belongs to the span in (32). The cardinal vectors prove surjectivity. Independence of the displayed basis follows by examining the coefficient of each nonmultilinear monomial.

For the eight-column core write $r = (r_1, \ldots, r_L)$ for the non-Boolean challenge path and $C = (r_2, r_3, r_4)$. Project the witness onto its first four native coordinates. The gate queries give (r_4, b) for $b \in \{0,1\}^3$. The copy and public-input projections are $(0, C)$ and (r_1, C). All ten points are distinct even if every challenge has the same value. For each tree, projection to the first and last coordinates gives $(1, r_L), (0, r_L), (r_2, 0), (r_2, 1), (r_1, r_L)$.

Their five coordinate states are distinct. The independent tree masks therefore make all eight boundary values and the two terminal values jointly uniform. Keeping the eight boundary values and subtracting the terminal values is a surjective map onto the nine-dimensional joint interface.

Counting the nonmultilinear exponents in (32) gives $|\mathcal{B}_{4,10}| = 348$ and $|\mathcal{B}_{2,5}| = 21$. Thus this particular construction uses $348 + 21 + 21 = 390$ independent source coefficients. Construction 1 and the core witness mask use $10 + 5 + 4 = 19$ for the protocol's prescribed observations. The generic guarantee applies to distinct projections. Two complete points with identical projections receive the same mask value. The norm–trace alternative in Appendix A.7 gives different coefficient counts and degree bounds for the same projected query budgets.

Appendix B Security Proofs

B.1 Conditional message distributions

Fix the public parameters, their simulation trapdoor (τ, ζ) with $\zeta \neq 0$, and a complete challenge function H with legal outputs. Fix a classical witness branch. The following four calculations give the handlers in Algorithm 1.

Commitments. For every fixed source or helper P, the map $r_P \mapsto c = P(\tau) + \zeta r_P$ is a bijection of $\mathbb{F}$. Conditional on c, its unique preimage is $(c - P(\tau))/\zeta$. Thus every possible P gives c the same probability. This remains true when a tree polynomial depends on the witness and the previously sampled fingerprint. Fresh blinds yield the product distribution for several commitments. The simulator knows the public index polynomials, stores each private exponent it samples, and uses the prescribed parent combination for a derived handle.

Round messages. Fix source polynomials compatible with the reached history. Together with the public packet they fix F^*. The commitment calculation leaves the helper coefficients independent and uniform. Proposition 4.1 gives a fresh block scaled by $\rho 2^{L-a}$ at round a. The unconsumed blocks sum to zero on the Boolean suffix. The diagonal factor is nonzero. Hence the transmitted coefficients have the uniform law even after conditioning on every preceding message and the chosen source state. Conversely, the transcript and that source state determine the consumed coefficients successively. All compatible source states therefore assign the same probability to these messages. This leaves the source-mask posterior unchanged. The helper endpoint is then forced by (13).

Source observations. For a fixed branch s, let the existing joint constraints on the masks be $A_t R = z_t - b_t(s)$ at history t. The preceding two calculations leave R uniform on this fiber. Lemma 3.1 gives the next observation law

$$\operatorname{Unif}\big(b'(s,t) + L_t r_{s,t} + L_t \ker A_t\big). \tag{33}$$

Here $r_{s,t}$ is any solution of the previous constraints. The source maps in Section 3 make the independent observations jointly uniform for every legal choice of complete points.

Conditional on previous answers, their joint linear dependencies thus give an affine support with public coordinates. For example, the two trees contribute eight boundary values and one root difference. Reversing that difference gives its negative. Recording the logical handles, complete point, and joint functional preserves this dependence in later responses.

Opening messages. Freeze a commitment exponent c, complete point z, and evaluation y. In an honest opening $c = P(\tau) + r_P\zeta$, $y = P(z)$ and

$$a_i = Q_i(\tau) + \zeta u_i \quad (1 \le i \le L), \qquad a_0 = r_P - \sum_{i=1}^{L}(\tau_i - z_i)u_i.$$

For fixed P and its sequential quotients, each fresh uniform u_i gives an independent uniform a_i. Telescoping then gives the unique remaining coordinate

$$a_0 = \zeta^{-1}\left(c - y - \sum_{i=1}^{L}(\tau_i - z_i)a_i\right). \tag{34}$$

Algorithm 1 samples exactly this hyperplane. The same calculation applies to frozen linear combinations of commitments and values, including a zero aggregate. Fresh opening coins are required at every invocation, including a repeated point or a cached quotient. Optional post-opening batching applies the same function of the completed transcript in both experiments.

B.2 Classical–quantum composition

Fix H and the setup values. Let S denote the classical secret branch, including selected witnesses and any classical side labels. Write R for private prover coins and E for the environment's quantum register. At every positive-probability public history t, the real experiment has the form

$$\rho_{SRE|t} = \sum_s p(s \mid t)|s\rangle\langle s| \otimes \mathrm{Unif}(\mathcal{F}_{s,t})_R \otimes \rho_{E|s,t}. \tag{35}$$

Fixing s is essential because $\rho_{E|s,t}$ may depend on the witness. The fiber $\mathcal{F}_{s,t}$ is parameterized by the source fibers and unconsumed coins. The consumed blinds, helper coefficients, and opening coins are recovered from the messages and these parameters. This parametrization can be nonlinear because F^* can depend nonlinearly on the sources.

Proof (Proof of Theorem 5.2). Initially, fresh prover coins are independent of the joint witness and environment. Assume (35) before a response. The four calculations above give a common response kernel $K_t(y)$ on every branch s. Before discarding the secret label the joint external state after the response is

$$\sum_{s,y} p(s \mid t)K_t(y)|s, y\rangle\langle s, y| \otimes \rho_{E|s,t}.$$

Algorithm 1 induces the same state. Conditioning on y restricts each branch to a uniform subfiber. Between messages the environment applies the same quantum channels, including coherent calls to H. Measurements and scheduling decisions give the same classical branches. A fresh session adds independent coins. Induction over the finite event sequence proves equality at every message boundary. Tracing out S and averaging over setup and H preserves equality.

For an implementation with an independent private seed at total variation distance δ from ideal, data processing gives at most δ error over all its uses. For independent seed slots the bound is $1 - \prod_b (1 - \delta_b)$, charged separately on the real and simulated sides. Approximate adaptive kernels require a joint bound including the correlated reference register. The exact theorem uses ideal coins. The power discrete-log assumption for the classical extraction experiment does not provide quantum binding [79].

B.3 Typed evaluation consistency

Let Y be a formal hiding indeterminate. By the online representation interface, a frozen type 2 commitment has formal exponent $P(X)+rY$ with $P \in \mathcal{V}$, and a type 1 commitment has the same form with $P \in \mathcal{U}$. All representations are normalized to the actual basis lists, including reduction-available auxiliary elements. Public and derived handles use their designated interpretations. A role label alone does not restrict the exponent to an honest subspace.

Lemma B.1 (One global evaluation-consistency event). *In the registered experiment, let $b_U, b_V \geq 1$ be the actual total degree bounds and $\Delta = \max(b_V, b_U + 1)$. For all adaptively chosen openings checked by the enclosing verifier,*

$$\Pr[\textit{an opening accepts with } y \neq P_{\mathrm{frozen}}(z)] \leq \mathrm{Adv}^{\mathrm{powDL}}_{b_U, b_V}(\mathcal{B}) + \Delta/q + \epsilon_{\mathrm{alg}}.$$

The reduction is expected polynomial time when $\epsilon_{\mathrm{alg}} = 0$. The basis polynomials, their evaluation circuits, and their registered representations have polynomial size.

Proof. Choose the first accepted opening with an incorrect frozen evaluation. This event is recognizable by evaluating the known formal source polynomial at the public point z and checking the pairing equation. Normalize its proof elements as $Q_i(X) + u_i Y$ and $Q_0(X) + u_0 Y$, where $Q_i, Q_0 \in \mathcal{U}$. Acceptance implies vanishing at (τ, ζ) of

$$\mathcal{R}(X, Y) = P(X) + rY - y - \sum_i (X_i - z_i)(Q_i(X) + u_i Y) - Y(Q_0(X) + u_0 Y). \quad (36)$$

Its total degree is at most Δ, and it is nonzero because $\mathcal{R}(z, 0) = P(z) - y \neq 0$. The Y^2 term is covered by $b_U + 1 \geq 2$.

Embedding the challenge. Given the power-list challenge for unknown a, the reduction substitutes independently sampled affine functions of a for each coordinate of τ and for ζ. Specializing any basis circuit gives a univariate polynomial of degree at most b_U or b_V, so its encoding is computable from the respective power list. Reject and resample the

affine encoding of ζ if it is the identity. The resulting public setup has exactly uniform τ and uniform nonzero ζ. The reduction can run the prescribed indexer, adversary, random oracle, and the permitted auxiliary-input generator.

Recovering the exponent. Condition on a, the resulting (τ, ζ), the adversary's coins and transcript, and all its frozen representations. The hidden affine slopes remain independent uniform elements: the adversary sees only the resulting public encodings, not the embedding description. This is the point where the auxiliary-input restriction is necessary. For fixed nonzero $\mathcal{R}$, its highest nonzero homogeneous part evaluated on these slopes is a nonzero polynomial of degree at most Δ. Thus the probability that the specialized univariate residual is identically zero is at most Δ/q. Otherwise it is a nonzero univariate polynomial with a as a root. Factor it over $\mathbb{F}_q$ and identify a among its at most Δ roots using the challenge encoding $[a]_1$. This solves the power discrete-log instance. Specialization works directly on the compact basis circuits and does not require a dense multivariate expansion.

Selecting the first recognizable wrong opening gives one residual and one degeneration probability. More opening buckets increase the running time, not the coefficient of Δ/q.

For a strict running-time cutoff, cap nonzero-ζ rejection at u trials, with failure at most q^{-u}. For the specialized nonzero polynomial f, compute $\gcd(f, T^q - T)$ by modular exponentiation to isolate its distinct linear factors. Randomized splitting over an odd field, capped at v trials for each of at most Δ factors, contributes at most $\Delta(2/3)^v$ with the standard quadratic-character splitting rule: a random polynomial modulo a square-free factor has independent uniform values at two distinct roots, whose split classes disagree with probability at least $1/3$. Thus one available explicit choice is

$$\epsilon_{\mathrm{alg}} \le q^{-u} + \Delta(2/3)^v.$$

The extractor normalizes representations and evaluates Boolean restrictions. Only the reduction runs the factorization procedure.

B.4 From native relations to the extracted witness

The gate and public residuals directly impose (38) and (40). In the virtual profile, the latter holds by definition of the extracted witness. If leaf and internal residuals vanish, induction on the heap addresses shows that the canonical root is the product of its leaves. This argument uses only multiplication and also holds when a subtree product is zero.

Copy fingerprint. For a fixed extracted assignment, consider the bivariate polynomial

$$D_{\mathrm{cp}}(B, G) = \prod_I (w(I) + B\iota(I) + G) - \prod_I (w(I) + B\iota(\sigma(I)) + G).$$

It is identically zero exactly when the multisets of labeled values coincide. Unique factorization, with every factor monic in G and distinct integer labels in $\mathbb{F}_q$, then gives $w(I) = w(\sigma(I))$. If copying fails, $D_{\mathrm{cp}} \neq 0$. The degree-N homogeneous parts cancel

because the label multisets are equal. Also $D_{\mathrm{cp}}(0, G) = 0$, so B divides the difference. Hence D_{cp}/B is nonzero of total degree at most $N - 2$. As the sampled β is nonzero, the weighted Schwartz–Zippel bound gives (20)'s copy term. For $N = 2$ this polynomial is a nonzero constant, so its error bound is zero.

Lookup fingerprint. Remove the canonical zero-address factor G from the two $2n$-leaf products, which is allowed for the fingerprint argument because the challenge γ_{lk} is restricted to $\mathbb{F}_q^*$. The remaining left product consists of the $n - 1$ cyclic table-edge factors $t_j + Bt_{j+1} + G$ and n query self-loop factors $(1 + B)v(x) + G$. The right product consists of the $2n - 1$ edges of the nonzero H cycle. If the two formal products coincide, their monic-in-G linear factors give the same multiset of ordered value pairs.

Suppose a queried value u is absent from the physical table. The left edge multiset contains a loop (u, u), and every edge incident to u there is such a query loop. Thus the right cycle contains u, and its successor must again be u. Iterating around its single nonzero-address cycle makes every right-cycle value u. This is incompatible with the physical table's nonempty collection of edges, whose entries are not u. Therefore formal equality implies every query is a physical table member. This argument allows duplicate table values, including zero; the specially inserted zero-address loop has already been removed and does not act as a table entry.

If membership fails, the normalized product difference is a nonzero polynomial in (B, G) of degree at most $2n - 1$. Independent decoded coordinates with atom bounds μ_q, μ_{q-1} give the lookup term of (20). Only one failed semantic category is selected before the fresh fingerprint answer: take the maximum of the copy and lookup bounds, not their sum. The source H must already be frozen at that answer, which is why its commitment precedes the fingerprint packet.

B.5 From a nonzero relation to an incorrect opening

At a parsable registered packet node, all source and tree polynomials and the helper are fixed. Write $A_j(\omega_j) = \sum_{x \in \{0,1\}^{m_j}} F_j(x)$ and $B = \sum_{x \in \{0,1\}^L} g(x)$. For a canonical root A_j is a constant, not a fresh random compression. The honest helper has $B = 0$, but its actual published group space permits a malicious helper with $B \neq 0$. Soundness must cover both cases.

Combination packet. If $B \neq 0$, fix all other packet coordinates. At most one nonzero ρ satisfies $\sum_j \lambda_j A_j + \rho B = 0$, giving loss μ_{q-1}. If $B = 0$ and a fixed Boolean residual is nonzero, its equality-weighted sum is a nonzero multilinear polynomial in at most h coordinates. A nonzero multilinear polynomial is nonzero on independent coordinates of maximum atom μ_q with probability at least $(1 - \mu_q)^h$. To see this, induct on the variables: when the leading coefficient is nonzero, a nonconstant affine polynomial has at most one root; if the variable is absent, drop it. Conditional on a nonzero vector (A_j), the affine-chart combination with $\lambda_1 = 1$ cancels with probability at most μ_q; if only its first coordinate is nonzero, cancellation is impossible. Consequently

$$\Pr[\text{packet hides the fixed defect}] \leq \max\{1 - (1 - \mu_q)^{h+1}, \mu_{q-1}\}.$$

One fixed failed residual suffices for this bound. The nonzero random ρ accounts for an adversarial helper with nonzero Boolean total.

Rounds. At round a, suppose the incoming claim differs from the true remaining Boolean sum of the frozen $F^* + \rho g$. The fixed transmitted round polynomial obeys the endpoint-sum constraint. Its difference from the true partial-sum polynomial is nonzero and has degree at most D_a, by (18). It can repair the false claim at the fresh non-Boolean challenge only on at most D_a roots, giving probability at most $D_a\mu_{q-2}$. For the first round, the stronger two-zero endpoint format still implies the required endpoint sum; malicious helper half sums need not individually vanish for this argument. Honest E_a controls what is sent; the difference polynomial is controlled by D_a.

Terminal claims. If no such repair occurs, the terminal sumcheck claim is false. After all terminal values have been fixed, either a claimed source linear functional is wrong, or all are correct and (13) is a wrong helper value. In the former case select one wrong bucket before its fresh weights. With coefficients $(1, \alpha_2, \ldots)$ a nonzero error vector cancels with probability at most μ_q; a singleton never cancels. If it does not cancel, accepting that bucket is an event in Lemma B.1. In the latter case the singleton type 1 helper bucket is directly such an event. Selecting one wrong bucket avoids a factor equal to the number of buckets.

B.6 Fiat–Shamir contexts, reachable histories, and survival

Each challenge is the decoding of a domain-separated oracle output whose input canonically binds the version, SRS and actual basis identity, profile, coordinate anchor and maps, both E and D, index, statement, proof identifier, stage, group types and roles, ordered complete points, frozen values, message lengths, and the preceding transcript and challenges. Packet components use disjoint raw randomness. At the same-point stage the entire bucket list and derived helper value are included. At the optional pairing stage all opening bytes are additionally included. Verification rejects out-of-profile lengths, invalid encodings or subgroups, inconsistent parent-derived handles, and invalid state-machine transitions.

For the analysis, define the bad set at every fresh query before its answer. On a parsable registered prefix, evaluate the frozen formal polynomials and choose a failed semantic condition, a false claim, or a terminal error vector when appropriate; otherwise choose the empty set. On an unparseable or unregistered node also choose the empty set. Such a node can lead to an accepted proof only by a separate registration, context, or state-chain inconsistency. This definition depends only on the reached history. Future PCS errors or hash collisions are accounted for separately. All local bounds above hold at every reachable history to which the corresponding nonempty bad set applies.

Lemma B.2 (Adaptive budgeted survival). *Suppose a bounded online process chooses the category c_t of each fresh trial and its bad set from the past. At every reachable*

history the conditional bad probability is at most $\delta_{c_t} < 1$, and category c is used at most n_c times. Then

$$\Pr[\textit{some trial is bad}] \le 1 - \prod_c (1-\delta_c)^{n_c}.$$

No independence assumption on the bad events is needed.

Proof. Let S_t be survival through t trials, $N_c(t)$ the number of category-c trials used, and

$$d_t = \prod_c (1-\delta_c)^{N_c(t)}, \qquad M_t = \mathbf{1}_{S_t}/d_t.$$

If the next conditional failure probability is $p_t \le \delta_c$, then $\mathbb{E}[M_{t+1} \mid \mathcal{F}_t] = \mathbf{1}_{S_t}(1-p_t)/(d_t(1-\delta_c)) \ge M_t$. At the bounded stopping time T, $\mathbb{E}M_T \ge 1$ and $d_T \ge d_{\min} = \prod_c (1-\delta_c)^{n_c}$. Therefore $\Pr[S_T] = \mathbb{E}[d_T M_T] \ge d_{\min}$. If some bound is at least one, use the trivial failure bound one. Independent Bernoulli trials exhausting all budgets attain equality, so this bound cannot be improved using only these premises.

Apply the lemma with the distinct-query budgets, plus one final verification query of each category and each sumcheck depth. A repeated oracle input is not a fresh trial: its previously fixed answer is already accounted for. Abandoned branches and permitted auxiliary computations remain in the budgets. Outside the PCS, context, and chain events, any accepted invalid extracted witness must encounter one of these unconditionally defined bad sets. Taking a union with the base events yields the sharper form of Theorem 5.3. With all local probabilities truncated at one, it reads

$$\begin{aligned}\Pr[\text{knowledge failure}] \le \epsilon_0 + 1 - (1-\theta_{\mathrm{fp}})^{Q_{\mathrm{fp}}+1}(1-\eta_{\mathrm{pkt}})^{Q_{\mathrm{pkt}}+1} \\ \cdot \prod_{a=1}^{L} (1-\min\{1, D_a\mu_{q-2}\})^{Q_a+1}(1-\theta_{\mathrm{batch}})^{Q_{\mathrm{batch}}+1}.\end{aligned}$$

Here ϵ_0 is given in (21). Applying the union bound to the local factors gives the additive form in (22).

For full injective contexts no digest-collision term is needed. If three domain-separated digest layers of length κ_{ctx} replace them, let C_j count distinct digest queries and A_j count downstream references, including references fixed before their claimed preimages are bound. The usual collision and prebound-target events give the explicit ledger

$$\epsilon_{\mathrm{ctx}} \le \sum_{j=0}^{2} \left[\binom{C_j}{2} + C_j A_j \right] 2^{-\kappa_{\mathrm{ctx}}},$$

$$\epsilon_{\mathrm{chain}} \le \left[\binom{C_{\mathrm{tr}}}{2} + C_{\mathrm{tr}} A_{\mathrm{tr}} \right] 2^{-\kappa_{\mathrm{tr}}}.$$

These bounds assume independent random-oracle digest layers and count preprocessing and final verification. Full-prefix encoding has zero chain-compression loss.

If only $Q_{\mathrm{sc}} = \sum_a Q_a$ is known, put $\theta_a = \min\{1, D_a\mu_{q-2}\}$ and $\theta_{\max} = \max_a \theta_a$. Then

$$\prod_a (1-\theta_a)^{Q_a+1} \ge (1-\theta_{\max})^{Q_{\mathrm{sc}}} \prod_a (1-\theta_a).$$

Linearizing retains $Q_{\mathrm{sc}}\theta_{\max} + \sum_a \theta_a$. This retains the actual degree of each round on the final verification path.

B.7 Aggregation and post-opening verification

For a bucket with frozen nonzero error vector $\delta \in \mathbb{F}_q^r$, the cancellation event is $\langle A, \delta\rangle = 0$. With $A = (1, a_2, \ldots, a_r)$ and independent uniform tail coordinates, its probability is $1/q$ if the tail of δ is nonzero, and zero otherwise. For $r \geq 2$ no distribution confined to this affine chart improves its worst-case value: averaging over all errors with nonzero tail gives exactly $1/q$ for every fixed coefficient vector. For $r = 1$, coefficient one never cancels an error.

Proposition B.3 (Projective minimax cancellation). *Over all distributions of a single coefficient vector, the minimum worst-case cancellation probability on $\mathbb{F}_q^r \setminus \{0\}$ is*

$$\rho_r = \frac{q^{r-1} - 1}{q^r - 1}.$$

If the error lies in a publicly certified r-dimensional subspace and $t \leq r$ linearly independent checks are actually verified, the corresponding minimax value is

$$\rho_{r,t} = \frac{q^{r-t} - 1}{q^r - 1}.$$

Uniform nonzero directions attain the first value; a uniformly chosen t-dimensional row space attains the second.

Proof. Every nonzero coefficient row has $q^{r-1} - 1$ nonzero errors in its kernel. Averaging over uniform nonzero errors gives ρ_r for any fixed nonzero row, and choosing the zero row can only increase this average. Hence every distribution has a worst-case error at least ρ_r. Uniform nonzero rows have that cancellation probability for each fixed nonzero error, attaining the bound. Identifying scalar multiples gives the same event on projective space. For rank-t row spaces, the common kernel has dimension $r - t$, hence $q^{r-t} - 1$ nonzero vectors. The same average and symmetry argument proves the second assertion. Allowing rank below t enlarges the kernel and cannot improve the minimax value.

The bound for t checks requires a uniform t-dimensional row space and verification of all t equations. Public relations used to restrict the error space must be verified before sampling its dual coefficient space.

For a canonical projective codec, normalize the first nonzero coordinate to one. There are $P_r = (q^r - 1)/(q - 1)$ directions and P_{r-1} bad directions for a fixed nonzero error. Reducing b bits modulo P_r therefore gives the bound $P_{r-1}\lceil 2^b/P_r\rceil/2^b$. This decoder uses the projective domain of size P_r. The packet mixing coefficients can likewise use this variant, replacing the first component of η_{pkt} by

$$[1 - (1 - 1/q)^h] + (1 - 1/q)^h \rho_J$$

under exact uniform decoding. The basic implementation profile uses the affine chart, whose simpler codec already gives the stated bound.

Finally consider (37). After every bucket's full opening, point, value, type, and commitment is frozen and subgroup-checked, let its multiplicative verification residual in $\mathbb{G}_T$ be R_b. Bilinear expansion shows that the single equation tests $\prod_b R_b^{\gamma_b} = 1$. If one R_b is nonidentity, fixing the other weights leaves at most one bad value of γ_b in the prime-order group. Uniform private weights therefore give probability at most $1/q$, and decoded independent field weights give μ_q. The verifier need not compute a target-group discrete logarithm to use this argument.

For deterministic Fiat–Shamir verification, the new domain-separated input binds all $\Pi_{b,0:L}$ in addition to the earlier endpoint context. Its bad set is now fixed before its answer, so Lemma B.2 adds the factor $(1-\mu_q)^{Q_{\mathrm{pair}}+1}$. Earlier same-point weights do not give this condition because the adversary can choose the opening elements afterwards. When individual equations are correct, aggregation is perfectly complete; otherwise this new factor accounts for the possibility that aggregation hides a failed equation. The combined equation uses at most $L+3$ pairing terms for the same $B(L+1)$ opening elements.

B.8 Security of the experimental controls

Consider the eight-column core7 configurations used in the paired experiments. The privacy statements use the fixed oracle, classical proof interface, same-distribution simulation setup, and independent uniform private coins of Theorem 5.2. The soundness statements use the classical registered algebraic experiment of Theorem 5.3. Each verifier binds its format, degree caps, and challenge schedule in its own transcript context.

Projected source masks (GG). Appendix A.8 proves surjectivity on the ten distinct witness projections and five distinct projections per tree along every legal path. Independent tree masks make all ten tree values jointly uniform. Retaining eight boundary values and one root difference is a surjective linear map onto the prescribed nine observations. The source handler of Algorithm 1 therefore applies on this interface. GG uses a fresh SRS for its enlarged source and quotient spaces. The degree limits b_U, b_V in (19) and the round bounds D_a in (22) must be computed for those published spaces.

A total-sum helper (QT). The additional uniform term $b(X_1 - \frac{1}{2})$ preserves the Boolean sum and belongs to the existing helper support. It adds $\rho 2^{L-1} b$ to the first-round linear coefficient. Conditional on the higher coefficients, this coefficient is uniform. The initial sum claim fixes the constant by $2h_{1,0} + \sum_{e\geq 1} h_{1,e} = 0$. Later rounds use the usual recurrence. The source and opening kernels, published SRS, and actual degree caps are those of QG. The soundness argument in Appendix B.5 needs only the endpoint-sum identity in the first round.

Separate native sumchecks (QM). Write d_j for check j's native dimension and $o_j = L - d_j$. Each check commits to an independently blinded helper before the joint packet supplies

its equality challenges and independent nonzero mixer ρ_j. At shared depth a, every active check fixes its round message before r_a is sampled. For check j at local round $i = a - o_j$, the fresh block is scaled by $\rho_j 2^{d_j - i}$. These blocks are jointly uniform conditional on the past and the source state. The derived endpoint of helper j is authenticated at the complete point $(0^{o_j}, r_{o_j+1}, \ldots, r_L)$. These are the same zero-prefix slices used to define the native sums, including for an arbitrary polynomial represented in the shared helper group.

For soundness, select one failed native relation before the packet reply. If its frozen helper has nonzero Boolean total on that slice, at most one ρ_j cancels the sum. For zero helper total, a failed equality-weighted residual has a nonzero multilinear compression polynomial, and a failed root gives a nonzero constant. With h the greatest equality-weighted native dimension, the packet loss is thus at most $\max\{\mu_{q-1}, 1 - (1 - \mu_q)^h\}$. At depth a a false claim for the selected check can be repaired at at most $D_{j,i}$ roots. Here $D_{j,i}$ covers the native expression, the transmitted cap, and all helper powers permitted on the slice by the shared SRS. A uniform depth bound is $\mu_{q-2} \max_{j:o_j<a} D_{j,a-o_j}$. With these packet and depth bounds and QM's own query budgets, Lemma B.2 applies. A surviving incorrect endpoint either cancels in its typed bucket with the stated batching loss or gives the single event in Lemma B.1.

Appendix C Full Protocol Specification and Implementation Details

C.1 Typed parameters and their use

Construction C.1. Typed Setup, Index, Commit, Batch, Open and Verify

Parameter input. A field, pairing groups, the ordered source and observation maps, degree certificates E, D, and the finite supports V, U.

1. **Setup.** Sample $\tau \leftarrow \mathbb{F}_q^L$, $\zeta \leftarrow \mathbb{F}_q^*$. Publish the source basis evaluated at τ in $\mathbb{G}_2$, the quotient and helper basis evaluated in $\mathbb{G}_1$, the hiding elements $[\zeta]_t$, the generators, and $[\tau_a]_2$ for verification.
2. **Index.** Check shape, gate, wiring permutation, public slice, and any lookup table and cycles. Compute the actual public polynomials, their type-2 commitments with zero blinds, and an index digest binding the profile and SRS identity.
3. **Commit.** For a registered private polynomial P, draw $r_P \leftarrow \mathbb{F}_q$ and compute $[P(\tau)]_t + r_P[\zeta]_t$. For a derived handle, take its prescribed linear combination of commitments.
4. **Batch.** Freeze the ordered evaluations from the terminal relation. Partition by (t, z) in order of first occurrence. Within each bucket sample $(1, \alpha_2, \ldots)$ and combine polynomial, blind, value, and commitment with the same coefficients.
5. **Open.** Compute the ordered telescoping quotients at z and use independent opening coins in (16). Send $\Pi_1, \ldots, \Pi_L$, followed by the correction Π_0.
6. **Verify.** Check canonical encodings, group and subgroup membership, and consumption of the entire input. Check (17) in every bucket, or check (37) with independent post-opening weights.

For a monomial Y^α, substitution of coordinates from first to last gives the quotient on axis a as

$$\Big(\prod_{i<a} z_i^{\alpha_i}\Big) \sum_{b=0}^{\alpha_a - 1} z_a^{\alpha_a - 1 - b} Y_a^b \prod_{i>a} Y_i^{\alpha_i}.$$

Thus $Q(V)$ is computable from the union of these monomial supports. A further quotient on the same axis decreases its exponent, and a quotient on a later axis has the support obtained directly at that axis of the source monomial. Earlier-axis quotients are zero. Quotients of pure powers are lower pure powers or constants. This proves closure of U in (15). All $[\tau_a]_1$ belong to U, so the prover computes the correction in (16) as $[r_P]_1 - \sum_a \xi_a([\tau_a]_1 - z_a g_1)$ using public operations.

The actual support used for soundness includes every direction made available under the same τ in its group. A wider previously issued SRS with that trapdoor enters the support union even if omitted from a new profile file. The narrow certificates assume fresh narrow parameters or a verified union-support calculation. Independent labels on handles do not restrict which public directions an algebraic prover may use.

Verification after freezing the opening bytes. For buckets b with frozen $(C_b, y_b, z_b, t_b, \Pi_{b,0:L})$, obtain new weights γ_b only after all opening bytes are fixed. Put

$$S_a = \sum_b \gamma_b \Pi_{b,a}, \qquad S_0 = \sum_b \gamma_b \Pi_{b,0},$$
$$T_0 = -\sum_{b,a} \gamma_b z_{b,a} \Pi_{b,a} - \sum_{b:t_b=1} \gamma_b (C_b - [y_b]_1), \qquad U_0 = \sum_{b:t_b=2} \gamma_b (C_b - [y_b]_2).$$

Replace the separate equations by

$$\prod_a e(S_a, [\tau_a]_2) e(S_0, [\zeta]_2) e(T_0, g_2) e(-g_1, U_0) = 1. \tag{37}$$

Bilinearity gives exactly the weighted product of the original residuals. As in the pairing batching used in cq [37], the verifier groups terms with equal second arguments and performs one final exponentiation. There are $L + 3$ pairing terms in the mixed-type case. The wire carries the complete openings in both verifier modes. The coefficients γ_b are drawn by a separate Fiat–Shamir stage or from fresh private verifier coins.

C.2 Indexed relation, scalar lookup, and boundary cases

Let $n = 2^\nu$, $k = 2^c$, $N = nk = 2^m$, and $m = \nu + c$. The index contains a gate polynomial f, public selector tables q, a permutation σ of $\{0,1\}^m$, and a public-input slice of length $2^{\nu_p} \le N$. The statement supplies its values p. The witness is a table $w : \{0,1\}^c \times \{0,1\}^\nu \to \mathbb{F}_q$. The relation requires

$$f(q(x), w(0,x), \ldots, w(k-1,x)) = 0 \qquad (x \in \{0,1\}^\nu), \tag{38}$$
$$w(I) = w(\sigma(I)) \qquad (I \in \{0,1\}^m), \tag{39}$$
$$w(0^{m-\nu_p}, b) = p(b) \qquad (b \in \{0,1\}^{\nu_p}). \tag{40}$$

Column integers in (38) denote their fixed c-bit encodings. The indexer checks dimensions, the permutation, the gate description, and the capacity of the selected parameter profile, and commits to the actual public multilinear polynomials with zero hiding scalar. The verifier authenticates this index.

The core ambient dimension is $L = m + 1$. With lookup it is $L = \max(m + 1, \nu + 2)$. A polynomial with $a < L$ native coordinates is used on its prescribed slice $(0^{L-a}, x)$. It may be stored as a polynomial independent of the leading coordinates, but the opening point always contains all L coordinates. Native notation below suppresses this fixed zero prefix.

The witness source has native coordinates (Z, C, X) and a Boolean restriction $W(0, C, X) = w(C, X)$. Its $Z = 1$ slice can be chosen publicly, for example as zero. The honest prover takes the multilinear extension and adds the vanishing witness mask specified in Section 3. The resulting actual committed polynomial is denoted P_W. The source masks preserve every Boolean table entry.

We use either of two public-input profiles, fixed before setup. In core7, the semantic witness is $\mathcal{W} = P_W$, and we include the native residual

$$R_{\mathrm{pub}}(Y) = \chi_{\mathrm{pub}}(Y)(P_W(Y) - p(Y_{L-\nu_p+1:L})), \qquad \chi_{\mathrm{pub}}(Y) = \prod_{a=1}^{L-\nu_p} (1 - Y_a). \tag{41}$$

It is evaluated on the native core slice. In core6, every witness occurrence, including lookup occurrences, instead denotes

$$\mathcal{W}(Y) = (1 - \chi_{\mathrm{pub}}(Y))P_W(Y) + \chi_{\mathrm{pub}}(Y)p(Y_{L-\nu_p+1:L}). \tag{42}$$

The public values override the stored slice in every witness occurrence. Extraction uses the Boolean restriction of (42).

Protocol C.2. Scalar Lookup Extension

Input. A core instance, a physical table $t_1, \ldots, t_{n-1}$, a scalar query per row, and two certified nonzero-address cycles.

1. Assign each Boolean query to a matching physical entry, count its multiplicity, and form the repeated sequence. Commit to its masked source P_H and the masked witness before either fingerprint.
2. Draw the copy challenges and $\beta_{\mathrm{lk}} \in \mathbb{F}_q$, $\gamma_{\mathrm{lk}} \in \mathbb{F}_q^*$ in one packet. Build the copy trees and the lookup trees with leaves (45) and (46). Commit to each masked pair.
3. Run Protocol 1 with the core residuals and the five lookup residuals below. Each root contributes its canonical difference. Every other residual receives a fresh multilinear equality weight.
4. Apply Construction C.1 at the full points below. Accept if all source and helper claims authenticate and the global sumcheck verifies.

The optional lookup index contains a nonempty physical table $t_1, \ldots, t_{n-1}$ and a registered scalar algebraic query function f_{lk}. The extra relation is

$$v(x) := f_{\mathrm{lk}}(q_{\mathrm{lk}}(x), w(0, x), \ldots, w(k - 1, x)) \in \{t_1, \ldots, t_{n-1}\} \quad (x \in \{0, 1\}^\nu).$$

Here $\nu \geq 1$. An honest prover chooses a matching physical address for each query, for example the first matching address, and lets a_j be the number of queries assigned to entry j. It forms the sequence that repeats t_j exactly $1 + a_j$ times, in physical-table order. This sequence has $2n - 1$ entries, including duplicate physical entries.

For each $d \in \{\nu, \nu+1\}$ the index fixes a certified primitive binary linear-feedback cycle S_d on $\{0,1\}^d \setminus \{0\}$, fixing 0. Its zero and one branches have the form

$$S_d(x_1, \dots, x_d) = \begin{cases} (0, x_1, \dots, x_{d-1}), & x_d = 0, \\ (1, x_1 \oplus a_1, \dots, x_{d-1} \oplus a_{d-1}), & x_d = 1, \end{cases}$$

with certified feedback bits. For evaluation over $\mathbb{F}_q$, each branch uses coordinates x_i or $1 - x_i$, and the branch selector is x_d. For a polynomial P define the explicit polynomial

$$\mathcal{S}_d P(x) = (1 - x_d) P(S_d^{(0)}(x_{1:d-1})) + x_d P(S_d^{(1)}(x_{1:d-1})). \tag{43}$$

Its Boolean restriction is $P \circ S_d$. No non-Boolean XOR operation is left implicit.

Place the physical table along the nonzero ν-cycle to obtain a public multilinear polynomial T, with $T(0) = 0$. Place the repeated sequence along the nonzero $(\nu+1)$-cycle. Sample the raw zero entry independently and commit to a masked source P_H with these nonzero Boolean entries; its raw entry at zero is irrelevant. Every protocol occurrence is the virtual source

$$\overline{H}(x) = (1 - \mathrm{eq}(x, 0^{\nu+1})) P_H(x). \tag{44}$$

Thus the semantic zero address is exactly zero without a private Boolean opening. The source mask for P_H is the three-direction space from Section 3. The witness uses the lookup version of its mask, since the source query family has changed.

Fix P_H before drawing lookup fingerprint coordinates $\beta_{\mathrm{lk}} \in \mathbb{F}_q$, $\gamma_{\mathrm{lk}} \in \mathbb{F}_q^*$. Let $d = \nu+1$. Construct two more division-free trees C, E with $2n$ leaves and native dimension $d+1$, using

$$\ell_C(b, x) = (1-b)(1+\beta_{\mathrm{lk}}) v(x) + b\big(T(x) + \beta_{\mathrm{lk}} \mathcal{S}_\nu T(x)\big) + \gamma_{\mathrm{lk}}, \tag{45}$$

$$\ell_E(y) = \overline{H}(y) + \beta_{\mathrm{lk}} \mathcal{S}_d \overline{H}(y) + \gamma_{\mathrm{lk}}. \tag{46}$$

The polynomial v uses the same semantic witness $\mathcal{W}$ as the core, at $Z = 0$ and the prescribed column addresses. The five additional residuals are precisely

$$\begin{aligned} &C(1, b, x) - \ell_C(b, x), \qquad E(1, y) - \ell_E(y), \\ &(1 - \mathrm{eq}(y, 0^d))\big(C(0, y) - C(y, 0) C(y, 1)\big), \\ &(1 - \mathrm{eq}(y, 0^d))\big(E(0, y) - E(y, 0) E(y, 1)\big), \\ &\mathrm{eq}(z_{1:d}, 0^d) z_{d+1} (C - E)(z). \end{aligned}$$

The first four are equality-weighted; the last is canonical. Trees C, E use another independent nine-dimensional joint mask. Adding these five residuals gives lookup11 or lookup12. Membership is tested against the physical positions. A zero query therefore requires a physical entry equal to zero.

For $c = 0, \nu \geq 1$, suffix alignment identifies the copy and lookup witness points and identifies the gate and public witness points, whenever both occur. Two coefficients on $\phi(Z)$ and $\phi(X_\nu)$ suffice. For $\nu = 0, c \geq 1$ the core has no row coordinate and may

use the ordinary column anchor. Recompute support after identifying coincident tree and witness axes. Lookup requires $\nu \geq 1$. At $N = 1$ the copy trees and root residual disappear, and the witness needs only $\phi(Z)$. The prover emits the two canonical identity tree commitments retained in the core encoding. The verifier checks both. These profiles and the joint interpolation arguments are detailed in Appendix A.

C.3 Deterministic separation of core opening points

For a nondegenerate core configuration, assume $c, \nu \geq 1$ and let $r \in (\mathbb{F} \setminus \{0,1\})^L$. Let $\mathcal{C}$ be the set of active gate columns, $A = |\mathcal{C}|$, and let s count distinct selector handles, including the output selector. All source-group points fall in the following classes. Each point is written in the common L-coordinate space.

Lemma C.1 (Core point separation). *The $A + 6$ source-group points in Table 4 are pairwise distinct for every legal challenge path. This requires neither distinct round challenges nor a restriction on whether two challenges sum to one.*

Proof. Different G_a points differ in a Boolean column bit. The second coordinate distinguishes S from P_0, because $c \geq 1$ and r_2 is non-Boolean. The first coordinate distinguishes P_1 from both. The first coordinate of G_a, I_0, I_1, R is non-Boolean, separating these points from S, P_0, P_1. The second coordinate of G_a is Boolean, whereas that of R is r_2. Since $\nu \geq 1$, both G_a and R have non-Boolean last coordinate r_L, separating them from I_0, I_1. Finally, I_0 and I_1 have different Boolean last coordinates. These comparisons cover all pairs.

The helper also opens at R, but belongs to the other group and therefore has its own bucket. Witness observations number $A + 2$ for core7 and $A + 1$ for core6. Each copy tree contributes four observations; their derived difference contributes one; the permutation contributes one; and the selectors contribute s. Thus

$$Q_7 = A + s + 12, \qquad Q_6 = A + s + 11, \qquad B_7 = B_6 = A + 7.$$

The count follows from the fixed source registry. A value that happens to be zero does not delete a query or change the encoding.

Full points with lookup. Let $\mathrm{pad}(x) = (0^{L-|x|}, x)$, $d = \nu + 1$, $u = r_{L-\nu+1:L}$, $v = r_{L-d+1:L}$, and $s = r_{L-d:L}$. For the core component, apply its native maps with the corresponding suffixes of r, then zero-pad on the left. For each active lookup column b, query P_W at $\mathrm{pad}(0, b, u)$. Query the public table at

$$\mathrm{pad}(u), \quad \mathrm{pad}(S_\nu^{(0)}(u_{1:\nu-1})), \quad \mathrm{pad}(S_\nu^{(1)}(u_{1:\nu-1})).$$

Query P_H at $\mathrm{pad}(v)$ and at $\mathrm{pad}(S_d^{(j)}(v_{1:d-1}))$ for $j = 0, 1$. Multiply each returned scalar by its own public zero-address factor from (44). Each $T = C, E$ uses

$$\mathrm{pad}(1, v), \quad \mathrm{pad}(0, v), \quad \mathrm{pad}(v, 0), \quad \mathrm{pad}(v, 1).$$

Table 4. Complete source-group point classes for the nondegenerate core.

Class	Full point	Uses
G_a, $a \in \mathcal{C}$	$(r_{c+1}, \mathrm{bits}(a), r_{c+2:L})$	Gate witness values
S	$(0^{c+1}, r_{c+2:L})$	Public selectors
P_0	$(0, r_{2:L})$	Copy witness, permutation, tree parents
P_1	$(1, r_{2:L})$	Tree leaves
I_0	$(r_{2:L}, 0)$	Left tree children
I_1	$(r_{2:L}, 1)$	Right tree children
R	r	Root difference; native public witness

The lookup root check queries $C - E$ at $\mathrm{pad}(s)$. In both core and lookup the global helper opens at r in $\mathbb{G}_1$. The implementation registers each ordered handle and full point once. Distinct logical handles at the same group and full point share an opening but contribute separate evaluation claims.

Lookup and degenerate configurations require their own full-point calculation. In particular, non-Boolean challenges alone do not rule out all lookup collisions. In the zero-based notation of a single-column example with $\nu = 2, L = 4$, the points $(0, 1, r_2, r_3)$ and $(0, 1, 1 - r_1, r_2)$ coincide when $r_1 = 2$ and $r_2 = r_3 = -1$. The canonical lookup registry merges coincident queries. The audit therefore distinguishes a certified all-path count, a symbolic general position upper bound, and an exact replay of the challenges in a full proof.

C.4 Degree and encoding certificates

For the nondegenerate query-aware core let $s_a = \max(1, \deg_{Y_a} V)$ denote the largest coordinate degree in the registered source basis, numbering axes from one. In the eight-column example these degrees are two on the witness slice axis, column anchor, and last axis and one on the remaining axes. Put $v = 1$ for a virtual public slice and $v = 0$ for an explicit check, and $b_a = \mathbf{1}_{a \le L - \nu_p}$. For the implemented gate families of witness degree δ and multilinear selectors, the native cap certificates, including their equality weights, can be reconstructed as follows.

$$\begin{aligned}
\text{Gate:} &\quad \big(\delta(s_a + v b_a) + 2\big)_{a \in (1, c+2, \ldots, L)}, \\
\text{Explicit public check:} &\quad (s_a + 1 + b_a)_{a=1}^{L}, \\
\text{Copy leaf:} &\quad (s_a + 1 + v b_a)_{a=2}^{L}, \\
\text{Internal node:} &\quad \big(\max(s_{a+1}, 2s_a) + 2\big)_{a=1}^{L-1}, \\
\text{Root:} &\quad (s_a + 1)_{a=1}^{L}.
\end{aligned}$$

For example, an internal child product has two factors at its leading variable, while the parent map shifts that variable by one axis. The zero-address exclusion and equality weight each add one. For each native cap vector insert leading ones to account for the prefix selector in (6). Their axiswise maxima bound F_*. Include actual helper degrees according to (18). The registered $E = D$ core uses these maxima as sending caps. With

tight honest caps and $c, \nu \geq 1$, the honest witness has last-row degree one before public virtualization, so the last sending cap becomes $\max(4, \delta(1 + v\mathbf{1}_{\nu_p=0}) + 2)$. This tightening affects neither the degree of a malicious source using the full span nor the need to charge it in soundness. With lookup, map the core expressions into their prescribed slices, include the two lookup leaf expressions and the second tree pair in the maximum, and recompute the support union. The implemented first-column and sum-column lookup functions are linear in their witness inputs.

Proof bytes. The core and lookup format has five header bytes followed by a fresh 32-byte salt. In order it carries C_W, optional C_H, C_A, C_B, optional C_C, C_E, C_g, the round payloads, the ordered source values, and the bucket openings. Its registered order of core checks is gate, optional public-input, copy leaves A, B, internals A, B, and root. Lookup appends its checks in the order of Protocol C.2. Each source list follows the indexed expression order. Retain a repeated handle/point only at its first occurrence. Bucket order likewise follows the first occurrence of each type and full point. In this implementation $\Pi_1, \ldots, \Pi_L$ precede the final correction Π_0. Proofs with truncated objects or trailing data are rejected.

Scalars have a 32-byte big-endian canonical encoding below q. Compressed $\mathbb{G}_1$ and $\mathbb{G}_2$ elements use 48 and 96 bytes. For $\mathbb{G}_2$, encode the imaginary then real component of the x-coordinate. The decoder checks field canonicality, compression and infinity flags, the y-sign choice, curve and subgroup membership, and canonical re-encoding. The unique infinity encoding begins with hexadecimal `c0` followed by zeros. An opening contains all $L + 1$ elements even when some coordinates or source evaluations happen to be zero.

The Fiat–Shamir context includes the protocol and curve, profile digest, dimensions, degree vectors, anchor, index commitments and digest, statement, SRS identifier, root observation convention, and opening verification stage. The salted history frames each event by its order, stage label, length, and payload. Each challenge request names its coordinate domains in order. For each coordinate, decode a 64-byte SHAKE256 output word u as $d + (u \bmod (q - d))$, with $d = 0, 1, 2$ for the field, nonzero, and non-Boolean domains. Record the response before the next event. The global helper evaluation follows from fields already bound in this history. The pairing packet binds the concatenated completed openings. Private masks and opening coins instead use canonical rejection sampling.

SRS bytes. Write $M = 2^L$. The source basis contains M Boolean Lagrange directions and the nonmultilinear monomial support of its masks. The helper-group basis contains the $M/2$ multilinear suffix basis, the quotient directions and the pure powers of (15), deduplicated. In the serialized public file, the fixed 88 bytes comprise the eight-byte format identifier, six eight-byte parameter fields, and a 32-byte digest. The two basis tables include their hiding elements. A separate L-point type-2 verification-key block accounts for $96L$. Let $e_V = \dim V - M$ and $e_U = \dim U - M/2$ count the actual extra

basis directions after deduplication. The serialized SRS length is

$$\begin{aligned} |\mathrm{SRS}| &= 88 + 96(\dim V + 1) + 48(\dim U + 1) + 96L \\ &= 120M + 96e_V + 48e_U + 96L + 232. \end{aligned} \tag{47}$$

For the usual core profiles with $c, \nu \geq 1$, $\dim V_7 = M + k + 6$, $\dim V_6 = M + k + 5$, and $\dim U = M/2 + \sum_a E_a - L + k + 2$. The general implementation constructs the support union explicitly, including lookup and coinciding-axis cases. Derived caches are not serialized and do not change the SRS digest. At $N = 1$, the current wire format retains two identity tree commitments, although the semantic copying protocol has no trees; it samples no tree masks or blinds and checks these identities.

Fixed gate example. The eight-column, degree-five family in Section 6.2 has tight sending caps $E = (4, 6, 4, 12, 7, \ldots, 7)$ with $m - 3$ trailing sevens. Setting $E = D$ changes the last seven to twelve. The tight profile gives 21,877 bytes at $N = 2^{20}$. Setting $n = 2^{20}$ rows with $k = 8$ instead gives $m = 23$, and hence 24,709 bytes, or 24,869 bytes at $E = D$, by substitution in (23). The last two lengths are analytical.

C.5 Deriving SRS caches without the trapdoor

The public main basis consists of Boolean Lagrange evaluations. An honest prover often needs multilinear monomials on a small subset of coordinates, for example column masks and their products with the last coordinate. Computing every such direction by a separate scan of the main basis repeats large sums. We instead marginalize once and apply the standard superset zeta transform within the selected coordinates.

For the following implementation formula only, use zero-based coordinates and set $r_0 = L - \nu$ and $\mathcal{S} = \{0, \ldots, r_0 - 1\} \cup \{L - 1\}$, with duplicate axes removed. For $\mathbb{G}_2$, the main points are $P_b = [\mathrm{eq}_L(\tau, b)]_2$ for $b \in \{0,1\}^L$. For $\mathbb{G}_1$, the main basis is the $(L-1)$-variable suffix basis, using $\tau_{2:L}$ and omitting global axis zero from the selected set. Thus its main table has $M/2$ points. In either respective domain, with P_b denoting that domain's basis point, compute

$$A_u = \sum_{b : b|_{\mathcal{S}} = u} P_b, \qquad Z_a = \sum_{u \supseteq a} A_u = \left[\prod_{i \in \mathcal{S} : a_i = 1} \tau_i \right]_j . \tag{48}$$

The identity follows by summing the Boolean Lagrange factors over omitted coordinates, then over unconstrained selected bits. The transform uses only additions of already published points. It neither reveals τ nor extends the actual public span. High-degree extras continue to use their original basis entries.

Let $s_0 = |\mathcal{S}|$, $B_V = 2^{s_0}$, and $B_U = 2^{s_0 - 1}$. Initializing each accumulator with its first point gives exactly

$$(M - B_V) + s_0 B_V/2 \quad \text{and} \quad (M/2 - B_U) + (s_0 - 1) B_U/2$$

additions for the two selected-coordinate transforms. Pure row directions outside $\mathcal{S}$ require another $n - \nu - 1$ additions in each group. For $c, \nu \geq 1$, the totals become

$$\mathbb{G}_2 : \; M + 2kc + n - \nu - 1, \qquad \mathbb{G}_1 : \; M/2 + k(c-1) + n - \nu - 1. \tag{49}$$

The resulting cache derivation costs $O(M + k(c+1))$ group additions and $O(k + L)$ temporary group points. This is the cache workspace, not the memory bound of the whole SRS or prover. These counts exclude checked decoding, normalization, hashing, and the existing suffix-basis construction, which uses $M - 1$ and $M/2 - 1$ additions. The SRS loader retains all canonical and subgroup checks and reads the main points in bounded chunks.

C.6 Caching quotient commitments

For a fixed immutable dense table, the sequential quotient at coordinate a depends on the preceding coordinates $z_1, \ldots, z_{a-1}$, but not on z_a or the remaining suffix. The commitment to that unblinded quotient can therefore be reused across compatible openings. The cache key binds the table identity, SRS identity, embedding offset, and preceding coordinates. It does not identify sources merely because their current values or byte encodings happen to agree.

Only deterministic quotient commitments are cached. Each opening samples new independent blinding coefficients, and its resulting $L + 1$ messages remain part of the complete proof. Regression fixtures use a fixed private random tape to check direct/cached byte equality; formal measurements use fresh operating-system randomness. The cache ablation in Section 7 therefore changes computation while preserving the measured proof length.

C.7 Randomness and expression evaluation

Application-input seeds are public workload parameters. They are independent of the prover's masks, commitment blinds, opening randomization, and proof salt. Private field elements use rejection sampling of canonical 32-byte scalars. With independent exactly uniform candidate bytes, this procedure has an exactly uniform output distribution. The actual implementation uses OsRng; it does not establish an information-theoretic claim about the operating system's entropy source. Hash challenges instead use SHAKE256 output and a specified modular decoder. Their possible bias enters the soundness bound, while fixed-function privacy requires legal challenge domains and matching response kernels rather than uniform hash outputs.

The coefficient ledger provides a separate check on the private sampler. Let $v \in \{0, 1\}$ indicate a virtual public prefix, $\eta \in \{0, 1\}$ indicate lookup, and $\iota = \mathbf{1}_{N>1}$. The implemented witness mask samples

$$R_W = k + \mathbf{1}_{c>0}(2 - v) + \mathbf{1}_{c=0,\nu>0} + \eta \mathbf{1}_{c>0} k$$

field elements. Including the helper, tree masks, source blinds, and BL opening coefficients, the total number of successful private field samples is

$$R_{\text{field}} = R_W + R_g + 2 + 11\iota + 16\eta + BL. \tag{50}$$

The lookup term includes the raw random entry $P_H[0]$ in addition to the three geometric mask coefficients. A separate 32-byte salt is sampled. The ledger is neither the mask-space rank nor the number of candidate bytes read by rejection sampling; it does not claim minimal randomness for $P_H[0]$.

Gate cost is similarly recorded at a defined abstraction level. For a positive exponent e, put $\ell(e) = \lfloor \log_2 e \rfloor + 1$ and let $h(e)$ be its population count. The audited ark-ff 0.4.2 pow path starts at one and processes the leading bit, making $\ell(e)$ squaring calls and $h(e)$ multiplication calls. The ideal binary chain starting at the base instead uses $\ell(e)+h(e)-2$ combined operations. An exponent-one factor in the gate compiler is a source node and has no Pow node.

For the unsimplified native PowerCopy gate $(1 - Z)(q_0 W_0^{\delta} - W_1 - q_{\text{out}})$, with $\delta > 1$, the audited expression path has

$$(S, M_{\times}, A_{+}) = (\ell(\delta), h(\delta) + 5, 4).$$

For the fixed-thirteen-selector gate of degree five, the corresponding counts are $(24, 35, 14)$. These count source-level arithmetic calls, excluding the production of source values and all commitment work. They are not machine instructions or complete prover operation counts. Public-prefix virtualization also changes this expression: for a selector with P prefix factors, of which v_0 are variable, each replaced witness occurrence adds $(0, 2(P + v_0) + 3, 2v_0 + 2)$ in the current unsimplified compiler.

Evaluation scope. The construction targets $O(N)+\text{poly}(\log N)$ field and group operations under fixed width, fixed gate and lookup relations, bounded per-axis degrees, and structured evaluation with bounded work per processed table entry. Dense tables are processed and geometrically folded, while low-degree extras contribute their small-support terms. These conditions must be checked for the chosen evaluator. In particular, the current unsimplified virtual-prefix expression pays the $P + v_0$ cost above at each witness occurrence; with a fixed public-prefix length, $P = L - \nu_p$ grows with $\log N$. We therefore retain this parameter in the implementation cost and do not infer a uniform linear-time theorem for every implemented profile from the table-folding argument. Gate degree, term count, width, and security parameters are also charged explicitly when they vary. An MSM is not a unit-cost machine operation, and public-input processing depends on its actual length.

Appendix D Additional Experiments and Reproducibility

The Xeon Gold mechanism experiments contribute 90 matched three-arm proofs and 60 matched helper-control proofs. The separate E5 matrix contains 80 planned configurations, 77 completed configurations with 765 proofs, and two additional lookup controls with 20 proofs. The project README identifies the archived E5 run and its accepted analysis.

D.1 Mechanism comparisons and measurement boundaries

Each Gold proof occupies a fresh process pinned to physical cores 0, 2, 4, and 6. At each scale the primary study uses all six arm permutations over ten blocks, with each arm in each position three or four times. A separate paired run balances the two QG/QT orders five to five. QG is measured anew in the QT study. Its ten observations at a scale are distinct from the ten QG observations in the primary study. The runs fix relation, public input and witness within each block. The two contrasts in the primary study share their QG reference. The design measures each change at the reference level of the other mechanism. It has no arm that jointly applies GG's mask and QM's native checks.

Table 5 reports the three-arm observations. The numerator in each paired ratio is the alternative's proving time, and its denominator is QG on the same witness. The confidence interval resamples these paired ratios at each scale. Table 6 uses the same statistical unit in the two-arm study.

The additional successful sampler call in QT gives 268, 356, and 400 private field elements per proof, compared with 267, 355, and 399 in QG. Proof sizes for QT are 12,629, 16,405, and 18,293 bytes. All six arm/scale combinations in this paired study have 33 transmitted endpoint scalars and 15 opening buckets. In the independent native arm the 70, 98, and 112 univariate messages at the three scales use the certified native caps of their respective checks. They share the same 11, 15, and 17 challenge depths as QG. The individual cap arrays and the complete-point opening counts are in the primary protocol ledger.

Both studies measure proving and verification at internal pre-return timestamps. Proving includes encoding. Verification includes canonical decoding, subgroup tests, transcript reconstruction, and all cryptographic checks. Setup generation, checked parameter loading, indexing, and public input generation are excluded from proving time. Process peaks cover initialization, proving, and verification and use the maximum observed high-water mark or scheduler sample. The E5 process may contain several proofs. In a fresh Gold process the peak pertains to one complete job.

Quartiles use Type-7 interpolation at position $(m-1)p$ among m observations indexed from zero. The primary mechanism study uses bootstrap seed 20261117, the helper control

Table 5. Matched proofs on the Gold 6230. QG uses query-aware masking and global sumcheck. GG changes the mask. QM changes the sumcheck organization. P and peak are per-arm medians. Time ratios use QG as denominator.

$\log_2 N$	Arm	Round scalars	Proof (B)	P (s)	Peak (MiB)	Time ratio [95% CI]
10	QG	79	12,597	0.685	4.9	reference
	GG	208	16,725	1.145	8.2	1.674 [1.661,1.691]
	QM	245	19,349	0.679	5.2	0.995 [0.976,1.018]
14	QG	107	16,373	4.864	39.2	reference
	GG	236	20,501	6.814	42.6	1.400 [1.395,1.411]
	QM	337	25,557	4.644	39.4	0.961 [0.952,0.964]
16	QG	121	18,261	16.592	162.5	reference
	GG	250	22,389	24.621	146.4	1.481 [1.474,1.491]
	QM	383	28,661	15.603	162.6	0.942 [0.937,0.947]

Table 6. Paired helper control on the Gold 6230, ten witnesses per scale. P is the per-arm median prover time in seconds. Ratio columns are within-witness medians with 95% bootstrap intervals.

$\log_2 N$	QG P	QT P	QT/QG time	QT/QG process peak
10	0.698	0.705	1.015 [0.988,1.030]	1.0015 [0.9864,1.0314]
14	4.916	4.897	1.000 [0.995,1.009]	1.0005 [0.9959,1.0048]
16	16.840	16.805	0.999 [0.990,1.004]	1.0001 [0.9992,1.0022]

uses seed 2026111703, and the E5 matrix uses seed 20261116. Each uses 10,000 resamples for intervals of sample medians where indicated. Same-proof verifier order alternates in the E5 matrix and follows seed parity in the Gold mechanism study. The E5 matrix retained the first observation within each process, reused checked public parameters, and ran in blocks without a host reservation or fixed clock frequency. Within-input pairing does not control changes in background load across those blocks.

D.2 Configuration ledger and parameter coverage

Table 7 lists 77 complete main configurations and two complete controls. Table 8 separately lists the three other main configurations. Each short name is the exact configuration identifier with only its trailing hyphen and six-character hexadecimal suffix removed. These prefixes uniquely identify the corresponding main or lookup-control configuration in the artifact, together with its raw observations, configuration digest and provenance. m is the recorded valid/expected sample count, p the thread count, and P the median internal proving time. Status C denotes a complete configuration. $N = nk$ throughout.

Names and switches. The `degree-d` rows use PowerCopy degree d; `terms-t` rows use dense expressions with t terms at degree 5; `width-k` varies k at fixed N. The $k = 8$ width point is the `degree-5` row, not an additional run. The $k = 1$ row uses a single-column Boolean degree-2 gate, so it is not a pure width-only change relative to the other rows. These scans use $N = 1024$, four threads, anchor 3, and cached openings.

In a name `fF`, F is the profile flag: mask value 1 selects tight honest round bounds, value 2 compiles the public-input relation, and value 4 adds lookup. Thus flags 0/1 select core7, 2/3 select core6, 4 selects lookup12, and 6 selects lookup11. Flags 0/2/4/6 use $E = D$. A suffix `aA` specifies anchor A; `pJ` in a public-input row means public_log $= J$, not a thread count. The eight `profile` rows use the fixed 13-selector gate, $k = 8$, degree 5, and public_log $= 1$. `engineering-direct` uses the same profile as `profile-f1-a3`, with direct rather than cached quotient commitments.

Lookup names `tT` and `zZ` specify T distinct values and inclusion ($Z = 1$) or exclusion ($Z = 0$) of zero; the distribution is named explicitly. Those rows use PowerCopy degree 5, $k = 8$, anchor 3, and public_log $= 1$. The lookup public-input scan uses two distinct values, a uniform distribution, and includes zero. The non-lookup public-input scan uses the same PowerCopy gate. Equal seeds across lookup configurations need not give equal numerical witnesses because the target generator binds the configuration.

The baseline rows use the fixed 13-selector gate, degree 5, $k = 8$, public_log $= 1$, and four threads. The scale and thread rows use the same gate, degree and width,

with public_log = ν, core7, tight honest bounds, anchor 3, and cached openings. The four-thread settings at $N = 2^{12}$ and 2^{16} are the corresponding scale rows; this reuse is not an additional configuration or additional samples.

Table 7: All complete configurations: 77 main rows and two controls. Proof lengths are constant within each listed configuration.

Configuration	Status	m	N	p	P (s)	Proof (B)
Gate degree						
`degree-2`	C	10/10	2^{10}	4	1.089070	7637
`degree-3`	C	10/10	2^{10}	4	1.103364	7893
`degree-5`	C	10/10	2^{10}	4	1.052233	8437
`degree-9`	C	10/10	2^{10}	4	1.205126	9589
`degree-17`	C	10/10	2^{10}	4	1.299976	11893
`degree-33`	C	10/10	2^{10}	4	1.642883	16501
Dense-expression terms						
`terms-1`	C	10/10	2^{10}	4	1.113578	8437
`terms-2`	C	10/10	2^{10}	4	1.112328	9077
`terms-4`	C	10/10	2^{10}	4	1.180937	10357
`terms-8`	C	10/10	2^{10}	4	1.273142	12309
`terms-16`	C	10/10	2^{10}	4	1.305193	12565
`terms-32`	C	10/10	2^{10}	4	1.397328	13077
Width at fixed N						
`width-1`	C	10/10	2^{10}	4	0.957007	5845
`width-2`	C	10/10	2^{10}	4	1.227145	8565
`width-4`	C	10/10	2^{10}	4	1.151212	8533
`width-16`	C	10/10	2^{10}	4	1.100896	8341
`width-32`	C	10/10	2^{10}	4	1.274478	8245
Public-input scan						
`public-f1-p0`	C	10/10	2^{10}	4	1.091445	8437
`public-f1-p6`	C	10/10	2^{10}	4	1.074955	8437
`public-f1-p7`	C	10/10	2^{10}	4	1.057825	8437
`public-f1-p8`	C	10/10	2^{10}	4	1.047260	8437
`public-f1-p10`	C	10/10	2^{10}	4	1.057525	8437
`public-f3-p0`	C	10/10	2^{10}	4	1.134552	9653
`public-f3-p6`	C	10/10	2^{10}	4	1.001086	8693
`public-f3-p7`	C	10/10	2^{10}	4	1.002197	8533
`public-f3-p8`	C	10/10	2^{10}	4	1.000748	8533
`public-f3-p10`	C	10/10	2^{10}	4	1.022259	8533
Eight honest-bound/anchor profiles						
`profile-f0-a1`	C	10/10	2^{10}	4	1.288249	12661
`profile-f0-a3`	C	10/10	2^{10}	4	1.282986	12597
`profile-f1-a1`	C	10/10	2^{10}	4	1.327145	12501
`profile-f1-a3`	C	10/10	2^{10}	4	1.320058	12437
`profile-f2-a1`	C	10/10	2^{10}	4	1.514611	13717
`profile-f2-a3`	C	10/10	2^{10}	4	1.510696	13653
`profile-f3-a1`	C	10/10	2^{10}	4	1.472350	13557
`profile-f3-a3`	C	10/10	2^{10}	4	1.448921	13493
Direct-opening control						
`engineering-direct`	C	10/10	2^{10}	4	1.722286	12437
Lookup and its public-input scan						

Continued on the next page.

Table 7 continued.

Configuration	Status	m	N	p	P (s)	Proof (B)
`lookup-f4-t1-uniform-z1`	C	10/10	2^{10}	4	1.578504	18837
`lookup-f4-t2-uniform-z1`	C	10/10	2^{10}	4	1.628511	18837
`lookup-f4-t127-uniform-z1`	C	10/10	2^{10}	4	1.590321	18837
`lookup-f4-t127-skewed-z1`	C	10/10	2^{10}	4	1.585466	18837
`lookup-f4-t127-uniform-z0`	C	10/10	2^{10}	4	1.624237	18837
`lookup-public-f4-p0`	C	10/10	2^{10}	4	1.543202	18837
`lookup-public-f4-p6`	C	10/10	2^{10}	4	1.575801	18837
`lookup-public-f4-p7`	C	10/10	2^{10}	4	1.606621	18837
`lookup-public-f4-p8`	C	10/10	2^{10}	4	1.550105	18837
`lookup-public-f4-p10`	C	10/10	2^{10}	4	1.575017	18837
`lookup-f6-t1-uniform-z1`	C	10/10	2^{10}	4	1.672367	19893
`lookup-f6-t2-uniform-z1`	C	10/10	2^{10}	4	1.709573	19893
`lookup-f6-t127-uniform-z1`	C	10/10	2^{10}	4	1.746659	19893
`lookup-f6-t127-skewed-z1`	C	10/10	2^{10}	4	1.736957	19893
`lookup-f6-t127-uniform-z0`	C	10/10	2^{10}	4	1.709606	19893
`lookup-public-f6-p0`	C	10/10	2^{10}	4	1.755917	20053
`lookup-public-f6-p6`	C	10/10	2^{10}	4	1.592610	19093
`lookup-public-f6-p7`	C	10/10	2^{10}	4	1.560554	18933
`lookup-public-f6-p8`	C	10/10	2^{10}	4	1.550121	18933
`lookup-public-f6-p10`	C	10/10	2^{10}	4	1.516773	18933
Seven-check trace repair and geometry						
`baseline-trace-N7`	C	10/10	2^{7}	4	3.985581	45445
`baseline-cached-N7`	C	10/10	2^{7}	4	0.353778	9605
`baseline-trace-N9`	C	10/10	2^{9}	4	14.472483	58213
`baseline-cached-N9`	C	10/10	2^{9}	4	0.740884	11493
`baseline-trace-N11`	C	10/10	2^{11}	4	63.621070	70981
`baseline-cached-N11`	C	10/10	2^{11}	4	1.750421	13381
Complete scale configurations						
`scale-N12`	C	10/10	2^{12}	4	2.909293	14325
`scale-N14`	C	10/10	2^{14}	4	8.122813	16213
`scale-N16`	C	10/10	2^{16}	4	28.339183	18101
`scale-N18`	C	10/10	2^{18}	4	111.613201	19989
`scale-N20`	C	5/5	2^{20}	4	390.834386	21877
Additional thread configurations						
`threads-N12-p1`	C	10/10	2^{12}	1	6.743680	14325
`threads-N12-p2`	C	10/10	2^{12}	2	4.195739	14325
`threads-N12-p8`	C	10/10	2^{12}	8	2.226650	14325
`threads-N12-p16`	C	10/10	2^{12}	16	2.006377	14325
`threads-N12-p18`	C	10/10	2^{12}	18	1.937736	14325
`threads-N16-p1`	C	10/10	2^{16}	1	88.483842	18101
`threads-N16-p2`	C	10/10	2^{16}	2	48.668853	18101
`threads-N16-p8`	C	10/10	2^{16}	8	18.592120	18101
`threads-N16-p16`	C	10/10	2^{16}	16	13.601943	18101
`threads-N16-p18`	C	10/10	2^{16}	18	13.260588	18101
Additional lookup controls						
`lookup-control-f0`	C	10/10	2^{10}	4	0.978648	8597
`lookup-control-f2`	C	10/10	2^{10}	4	0.983336	9653

The $N = 2^{22}$ observation preceded an operator stop following the revised $N = 2^{20}$ cap. Its raw terminal status is `error` and exit code is -15 (SIGTERM), linked to the

Table 8. Main configurations excluded from primary performance results. P denotes a partial observation; U denotes unexecuted.

Configuration	Status	m	N	p	Observed P (s)	Proof (B)
`scale-N22`	P	1/5	2^{22}	4	1560.684314	23765
`scale-N23`	U	0/5	2^{23}	4	–	–
`scale-N24`	U	0/5	2^{24}	4	–	–

stop control record; it is not a proof rejection. It has no confidence interval and is not a complete configuration. The unexecuted $2^{23}/2^{24}$ records retain the earlier scale-cap provenance. The later measurement-stop record defines the effective final cap. Planned B1 interleaving, additional cache repeats, larger parameter checks, and the large-instance format diagnostic were not executed. No extrapolated time is substituted for any missing measurement.

The completed rows total 765 main and 20 control proofs. The one partial-scale proof and 324 throughput proofs bring the formal total to 1110 distinct proof hashes. Nine E1 fixtures raise the final jointly checked set to 1119 distinct hashes. Earlier development and B1 bridge fixtures are separate evidence, outside that count.

D.3 Scale dispersion and initialization

Table 10 gives the full quartile summaries underlying Table 9. The intervals are limited to complete configurations; the five-sample 2^{20} interval in particular should not be interpreted as a precise estimate of systematic uncertainty.

At $N = 2^{20}$ the five individual/aggregate same-proof verifier-time ratios have median 1.386566 and interval $[1.341244, 1.421010]$. Measured prover quartiles are 386.260100 and 392.064518 seconds. The single $N = 2^{22}$ proof in Table 8 took 1560.684314 seconds and occupied 23,765 bytes. No interval is computed from that observation.

D.4 Paired comparisons and format attribution

Table 13 gives quartiles of the per-input prover-time ratios. All comparisons have ten matched inputs per side. They retain the original blocked order. Their confidence intervals, reported in Table 12 for B1 and in Section 7.4 for caching, quantify paired sampling variation.

Same-backend trace comparison. B1 uses trace masks, shared prefix challenges, and seven native checks. Both implementations use the BLS12-381/arkworks backend, four threads, the thirteen-selector gate, degree five, and $\nu_p = 1$. Each has an SRS for its published support. The ten matched witnesses per scale were executed in blocks. The planned interleaved B1 repeats were cancelled before execution. Table 12 compares complete implementations. B1 is the project's trace-based repair baseline. The official HyperPlonk implementation was outside this experiment.

The eight profile rows isolate implemented switches, not arbitrary intermediate protocols. At their fixed gate and public-input parameters, switching to tight honest

Table 9. Complete scale measurements. m is the number of proofs; P, V_i, and V_a are median internal seconds for proving, individual verification, and aggregate verification. Intervals are 95% bootstrap intervals for the prover median.

N	m	P [95% interval]	V_i	V_a	Proof (B)	Peak (MiB)
2^{12}	10	2.909 [2.824,2.991]	.367	.223	14,325	12.238
2^{14}	10	8.123 [8.077,8.291]	.408	.256	16,213	39.496
2^{16}	10	28.339 [28.229,28.713]	.461	.291	18,101	172.398
2^{18}	10	111.613 [110.242,112.656]	.569	.365	19,989	605.023
2^{20}	5	390.834 [386.229,428.394]	.717	.514	21,877	2,226.988

Table 10. Internal prover seconds for the complete scale series.

N	Median	Q1	Q3	IQR	Median 95% interval
2^{12}	2.909293	2.834510	2.980636	0.146127	[2.824044,2.991391]
2^{14}	8.122813	8.097591	8.232097	0.134506	[8.077159,8.291444]
2^{16}	28.339183	28.241728	28.626538	0.384810	[28.228974,28.712592]
2^{18}	111.613201	110.428024	112.595892	2.167869	[110.242249,112.656485]
2^{20}	390.834386	386.260100	392.064518	5.804417	[386.229204,428.393581]

bounds reduces the proof by 160 bytes at either anchor, and anchor $1 \to 3$ reduces it by 64 bytes at each flag value. Each legal profile also changes its support/SRS as specified. Reducing the honest bound E does not reduce the adversarial degree bound D. No separate backend with a single maximum cap on every axis was measured.

The `public-f1` rows all have 8437-byte proofs. Compilation in the `public-f3` rows gives 9653, 8693, 8533, 8533, and 8533 bytes at public-input logarithms 0, 6, 7, 8, and 10, respectively. For lookup, the five table/distribution variants have constant lengths 18837 (flag 4) and 19893 (flag 6), while the flag 6 public-input scan gives 20053, 19093, 18933, 18933, and 18933 bytes. These observations concern the listed workloads; in particular, compilation changes the indicator polynomial and round caps as well as the number of native relations.

The trace baseline uses seven helpers and seven helper Boolean sums, whereas the global geometry format uses one helper whose Boolean sum is fixed to zero. The corresponding two format components differ by $6 \cdot 48 = 288$ and $7 \cdot 32 = 224$ bytes. Other differences include endpoint derivation, round scalar counts, masking, and query geometry. They can be itemized as communication components but were not measured as

Table 11. Separate initialization observations. Each row is one checked SRS load/cache reconstruction and one index certification, not repeated online samples; there was no page-cache flush.

N	SRS file (B)	Loading (s)	Index certification (s)
2^{12}	989992	0.926851	0.085524
2^{14}	3939880	3.643618	0.314451
2^{16}	15737128	14.088968	1.186962
2^{18}	62923816	58.392026	5.297278
2^{20}	251668264	230.066249	19.224173

Table 12. B1 versus geometry. Each side has ten matched inputs. P is median internal proving time; the ratio column summarizes per-input ratios, not a ratio of medians.

N	B1 P (s)	Geometry P (s)	Ratio [95% interval]	B1 (B)	Geometry (B)
2^7	3.986	.354	11.196 [10.885,11.616]	45,445	9,605
2^9	14.472	.741	19.520 [18.739,20.022]	58,213	11,493
2^{11}	63.621	1.750	36.350 [35.478,37.447]	70,981	13,381

Table 13. Per-input prover-time ratio distributions.

Numerator/denominator	Median	Q1	Q3	IQR
Direct/cached, $N = 2^{10}$	1.330308	1.307859	1.370796	0.062937
B1/geometry, $N = 2^7$	11.196085	10.993133	11.386647	0.393515
B1/geometry, $N = 2^9$	19.520439	18.849200	19.975061	1.125862
B1/geometry, $N = 2^{11}$	36.349507	35.664901	37.270488	1.605587

independent intermediate protocols; their timing effects cannot be recovered by allocating portions of the total B1 ratio.

D.5 All thread settings

Table 14 gives all 12 thread settings. Each speedup distribution consists of ten per-input ratios t_1/t_p; efficiency is speedup divided by p. The $p = 1$ entries are self-ratio references, not additional measurements. The artifact retains full quartiles, CPU costs, source identities, and all 120 paired records.

Table 14. Complete thread scan. Intervals are 95% bootstrap intervals of paired-ratio medians; P is median internal prover seconds.

N	p	P (s)	Speedup [95% interval]	Efficiency [95% interval]
2^{12}	1	6.744	1.00000 [1.00000,1.00000]	1.00000 [1.00000,1.00000]
2^{12}	2	4.196	1.62087 [1.56956,1.64633]	0.81044 [0.78478,0.82317]
2^{12}	4	2.909	2.33552 [2.26603,2.39505]	0.58388 [0.56651,0.59876]
2^{12}	8	2.227	3.01462 [2.96866,3.11928]	0.37683 [0.37108,0.38991]
2^{12}	16	2.006	3.33515 [3.23999,3.44488]	0.20845 [0.20250,0.21530]
2^{12}	18	1.938	3.50226 [3.37441,3.58032]	0.19457 [0.18747,0.19891]
2^{16}	1	88.484	1.00000 [1.00000,1.00000]	1.00000 [1.00000,1.00000]
2^{16}	2	48.669	1.81745 [1.80430,1.84316]	0.90872 [0.90215,0.92158]
2^{16}	4	28.339	3.12841 [3.07444,3.13891]	0.78210 [0.76861,0.78473]
2^{16}	8	18.592	4.76082 [4.74335,4.77926]	0.59510 [0.59292,0.59741]
2^{16}	16	13.602	6.45651 [6.38344,6.54409]	0.40353 [0.39897,0.40901]
2^{16}	18	13.261	6.64128 [6.59552,6.73146]	0.36896 [0.36642,0.37397]

Process CPU cost was also recorded around the caller's operation. Its sampling boundary differs from the internal prover timestamp; we therefore report it as cost rather than dividing it by the internal wall time and calling the quotient CPU utilization. For example, median CPU seconds at $N = 2^{16}$ are 88.4685 at one thread and 108.065 at 18

threads. The observed wall-time decrease thus does not imply a proportional decrease in total CPU work.

D.6 Batch throughput and concurrent delays

The throughput experiment has nine complete batches and 324 proofs. Each batch timer spans synchronized input generation, proving, aggregate verification, hashing, memory records, and worker completion. SRS loading, indexing, thread-pool initialization, and output file writes lie outside this interval. Each layout has three batches of 36 proofs and the same 108 application inputs. Table 16 supplements the rates in Table 15. Its Q1/Q3 and minimum–maximum ranges are descriptive summaries of three batches, not confidence intervals. CPU/wall is measured over the same batch boundary, unlike the distinct internal/caller boundaries of the thread experiment.

Per-batch records, all delay quartiles, process peaks, input matching, and execution hashes are included with the artifact's throughput study. No concurrent delay sample is counted as an independent throughput batch.

D.7 Correctness, byte audits, and reproducibility boundaries

The E1 profile/boundary set consists of `core6`, `core7`, `dense`, `fixed13`, `lookup11`, `lookup12`, `lookup_k1`, `n1`, and `nu0`. All nine full fixtures passed the independent Python individual and aggregate pairing checks, with nine negative cases rejected. The earlier trace bridge checked two complete protocols, including the fixed 13-selector gate, using the archived Python pairing implementation; these bridge fixtures are distinct from the nine-fixture accounting set. The 66 ordinary Rust tests, these independent checks, and the 28 final analysis tests serve different purposes and are not pooled into one cryptographic-security test count.

The independent format/SRS audit accepted 77 geometry JSONL files containing 756 samples: the complete geometry main cases, the valid partial-scale prefix, and the two controls. This set of 77 files is not the same set as the 77 complete main configurations, which also include the three trace configurations. The nine full E1 fixtures also passed their separate byte-ledger audit. The ledger rebuilds support, degree caps, query geometry, and byte counts; SRS accounting compares recorded file lengths and primary point counts. It does not revalidate pairings or authenticate the encoded group points in an SRS file. Those tasks have separate runtime checks.

The E5 archive binds the final counts, original stop history, and source and executable identities. Its tabular summaries retain verifier distributions, stage counters, CPU costs, memory maxima, and within-proof verifier comparisons. Full timing definitions and per-sample identities remain in the JSON summaries. The raw formal JSONL files record proof hashes, byte lengths, and runtime verification outcomes. They are not a store of all 1110 full proofs. Full proof/context hex is retained for the nine E1 fixtures and earlier bridge fixtures, with those different datasets identified explicitly. The accepted hash-deduplication result is a data-integrity check, not an additional security argument.

Table 15. Throughput under an 18-thread budget. Brackets show batch-level Q1 and Q3, based on three batches per layout.

Workers × threads	Proof/s median [Q1,Q3]	Batch seconds	Peak (MiB)
1 × 18	.453633 [.452176,.454897]	79.3593	21.918
3 × 6	1.02975 [1.02806,1.03417]	34.9600	32.309
18 × 1	2.07242 [2.05702,2.07394]	17.3710	112.926

Table 16. Batch-level throughput details; three batches per layout.

Layout	Proof/s min–max	Batch seconds median [Q1,Q3]	CPU/wall median
1 × 18	0.45072–0.45616	79.3593 [79.1394,79.6158]	4.8123
3 × 6	1.02638–1.03860	34.9600 [34.8110,35.0174]	8.8635
18 × 1	2.04162–2.07546	17.3710 [17.3583,17.5021]	17.5130

The E5 parameter scans use $N = 1024$, and the large repeated scale series uses one fixed gate, width, and profile. The later Gold controls in Section 7.2 supply randomized comparisons of source masking, sumcheck organization, and first-round compression. They are recorded separately from these E5 measurements.

D.8 Artifact organization and replay

The project README links the E5 results, both Gold mechanism archives, the implementation specification, and the digest manifests. Each archive identifies the exact source, dependency lockfile, executable, public keys, input configuration, and analysis version used for its experiment.

For the Gold experiments, unpack the evidence and its matching runtime, check the manifests, and run the corresponding analyzer on the raw records. The analyzer verifies the relation, public input, witness, proof length, and public-key identities within each matched block before summarizing the ratios. The helper archive reuses the three original QG keys. To replay either study, pass its run directory, its frozen verifier, and a new output file to the archived replay script. It rebuilds each indexed public instance, checks the proof digest, and runs individual and aggregate verification on the saved bytes. The archives contain all 90 three-arm proofs and all 60 helper-control proofs.

For the E5 matrix, the raw records retain proof hashes, lengths, input identities, and runtime verification outcomes. Reanalysis of those records reproduces the performance tables. Nine exported profile and boundary fixtures also retain the complete contexts and

Table 17. Descriptive concurrent per-proof delays, in seconds. Each layout contributes 108 correlated observations; brackets are Q1/Q3, not independent-replication confidence intervals. Proving here uses an outer API timer, distinct from the internal main-matrix timer.

Layout	Proving median [Q1,Q3]	Aggregate verification	Input-to-record median
1 × 18	1.9334 [1.8902,1.9906]	0.2275 [0.2171,0.2599]	2.1905
3 × 6	2.6237 [2.5985,2.6555]	0.2507 [0.2427,0.2606]	2.8956
18 × 1	8.2577 [8.1712,8.3551]	0.2720 [0.2633,0.2787]	8.5467

proofs for independent pairing replay. The artifact specifies which additional trace fixtures support cross-language checks. A new timing run requires the registered configurations, checked parameters, physical-core placement, timer boundaries, and fresh private proof randomness.