A note on zero-knowledge ring switching

Andrei Kochergin
andrei@oumuamua.dev

Abstract

We make the ring-switching technique of Diamond and Posen zero knowledge, resolving a problem left open in the literature. Over $\mathbb{F}_{2^n}$, our technique uses $n-1$ auxiliary masks and $n-1$ quadratic constraints, checked in zero knowledge by the underlying commitment scheme.

1 Introduction

The ring-switching technique of Diamond and Posen [DP24] converts linear claims about an array of bits into a claim about an array of elements of $L = \mathbb{F}_{2^n}$ (typically $n = 128$), each packing n of the original bits (Section 2). The technique is not natively zero knowledge: its prover message reveals information about the bits, a leak Diamond left for follow-up work [D25, Section 5]. Zero-knowledge implementations such as Binius [Binius] remove this leak by adding $O(n^2)$ auxiliary witnesses.

We use the trace operator to extract bits (Section 3) and its algebraic properties to pack the masking witnesses themselves (Section 4.2). This gives a zero-knowledge ring-switching reduction with only $n-1$ quadratic constraints on the witnesses (Section 4.3), independently of the witness size: 127 for $n = 128$. The prover's message is hidden by pads committed in advance. The reduction outputs a masked dot-product claim for a subsequent zero-knowledge argument.

The use of the trace to extract bits is not new. Soukhanov writes coordinates as sums of Frobenius powers x^{2^j} [S24], and Diamond and Posen use Frobenius powers of a basis to describe transposition on public data [DP24, Section 1.3]. What appears to be novel is the application of the trace to the committed masking witnesses.

Section 2 recalls ring switching, Section 3 collects the standard facts we use, Section 4 gives the construction, and Section 5 proves its correctness and security.

2 The Diamond–Posen ring-switching protocol

We now summarize the ring-switching technique from [DP24], which reduces multilinear claims over $\mathbb{F}_2$ to claims over an extension field $L = \mathbb{F}_{2^n}$. Assume a commitment scheme that supports multilinear openings only over L. The prover packs n bit witnesses into each committed element of L. Ring switching efficiently batches multilinear claims about the individual bits into a single dot-product claim about these field elements. Diamond and Posen further reduce the dot-product claim to a multilinear evaluation claim via sumcheck. In this paper, we stop at the dot-product claim. The subsequent sumcheck reduction can be made zero knowledge using standard techniques.

For simplicity, assume $n = 2^\kappa$. Fix once and for all a basis $\beta_0, \dots, \beta_{n-1}$ of L over $\mathbb{F}_2$. For $x \in L$, write $x[i] \in \mathbb{F}_2$ for the ith coordinate of x in this basis, so

$$x = \sum_{i=0}^{n-1} x[i]\beta_i.$$

Identify the indices $0, \ldots, n-1$ with $\{0,1\}^\kappa$. For $z \in L^\kappa$ and $w \in \{0,1\}^\kappa$, define the multilinear evaluation weight

$$\mathsf{eq}(w, z) = \prod_{i=0}^{\kappa-1} ((1 - w[i])(1 - z[i]) + w[i]z[i]).$$

All arithmetic below is in L, with bits viewed as elements of L. In particular, addition and subtraction coincide.

Packing conventions. Write d for the number of remaining Boolean coordinates after the κ packed coordinates. The prover holds an array of bit witnesses $\mathsf{w}[v, j] \in \mathbb{F}_2$, indexed by $v \in \{0,1\}^\kappa$ and $j \in \{0,1\}^d$. The corresponding packed witnesses are

$$\widehat{\mathsf{w}}[j] = \sum_{v=0}^{n-1} \beta_v \mathsf{w}[v, j], \qquad j \in \{0,1\}^d. \tag{1}$$

Having committed to $\widehat{\mathsf{w}}$, the prover claims that the multilinear extension of w evaluates to y at a public point $(r_{\mathrm{lo}}, r_{\mathrm{hi}}) \in L^\kappa \times L^d$. Using the same definition of eq on d coordinates, set

$$q_{\mathrm{lo}}[v] = \mathsf{eq}(v, r_{\mathrm{lo}}), \qquad q_{\mathrm{hi}}[j] = \mathsf{eq}(j, r_{\mathrm{hi}}).$$

The claimed equality is therefore

$$y = \sum_{v=0}^{n-1} \sum_{j \in \{0,1\}^d} q_{\mathrm{lo}}[v] q_{\mathrm{hi}}[j] \mathsf{w}[v, j]. \tag{2}$$

First round. The prover sends n field elements

$$\widehat{s}[v] = \sum_{j \in \{0,1\}^d} q_{\mathrm{hi}}[j] \mathsf{w}[v, j], \qquad 0 \le v < n. \tag{3}$$

These elements are sent in the clear. The goal of this paper is to mask them. See Section 4. The verifier checks

$$\sum_{v=0}^{n-1} q_{\mathrm{lo}}[v] \widehat{s}[v] = y. \tag{4}$$

It remains to relate the message $\widehat{s}$ to the committed packed witnesses $\widehat{\mathsf{w}}$.

Ring switching. Regard the bits of $\widehat{s}[v]$ as column v of an $n \times n$ binary matrix. Reading row u in the same basis gives the field element

$$t[u] = \sum_{v=0}^{n-1} \beta_v \widehat{s}[v][u]. \tag{5}$$

For the honest message (3), the maps $x \mapsto x[u]$ are $\mathbb{F}_2$-linear and $\mathsf{w}[v, j]$ is a bit, so

$$t[u] = \sum_v \beta_v \sum_j q_{\mathrm{hi}}[j][u] \mathsf{w}[v, j] = \sum_j q_{\mathrm{hi}}[j][u] \widehat{\mathsf{w}}[j]. \tag{6}$$

Diamond and Posen [DP24, proof of Theorem 3.5] prove that if (6) holds for every u and the first check (4) passes, then the original claim (2) holds.

Batching. We now batch the n checks (6) into a single check. After receiving $\widehat{s}$, the verifier samples $\alpha \leftarrow L^\kappa$, multiplies check u by $\mathsf{eq}(u, \alpha)$, and adds the left- and right-hand sides:

$$\sum_u \mathsf{eq}(u,\alpha) t[u] = \sum_{j \in \{0,1\}^d} \left(\sum_u \mathsf{eq}(u,\alpha) q_{\mathrm{hi}}[j][u] \right) \widehat{\mathsf{w}}[j]. \tag{7}$$

This is a single dot-product claim about the committed packed witnesses $\widehat{\mathsf{w}}$. All coefficients on the right-hand side are public.

3 Basic facts about binary fields

The main difficulty in making the ring-switching protocol zero knowledge is how to encode the transposition (5) into algebraic constraints. To this end, we need the standard definition of the (absolute) trace [LN, Section 2.3].

In characteristic two, $x + x = 0$, and squaring is additive: $(x+y)^2 = x^2 + y^2$. Repeated squaring gives the *Frobenius powers* x^{2^j}. For every $j \geq 0$,

$$(x+y)^{2^j} = x^{2^j} + y^{2^j}, \qquad (xy)^{2^j} = x^{2^j} y^{2^j}.$$

Moreover, $x^{2^n} = x$ in L. Consequently, for $0 \leq j < n$, a public coefficient can be moved inside a Frobenius power:

$$a x^{2^j} = \left(a^{2^{n-j}} x\right)^{2^j}. \tag{8}$$

The *(absolute) trace* is

$$\mathrm{Tr}(x) = \sum_{j=0}^{n-1} x^{2^j}.$$

It is $\mathbb{F}_2$-linear and satisfies $\mathrm{Tr}(x^2) = \mathrm{Tr}(x)$. Also $\mathrm{Tr}(x)^2 = \mathrm{Tr}(x)$, so $\mathrm{Tr}(x) \in \mathbb{F}_2$. For our fixed basis β, there exists a trace-dual basis γ satisfying

$$\mathrm{Tr}(\gamma_u \beta_v) = \begin{cases} 1 & u = v, \\ 0 & u \neq v. \end{cases}$$

By linearity of the trace, this recovers each coordinate:

$$x[u] = \mathrm{Tr}(\gamma_u x). \tag{9}$$

Equation (9) gives a closed form for extracting a bit from a packed word. We combine these expressions to implement the transpose.

4 Zero-knowledge ring switching

In order to achieve zero knowledge, we mask the message (3) with additive pads committed in advance. The first check (4) is affine in the pads and thus can be handled in zero knowledge by the commitment scheme. The main problem is how to verify the transposition (5) when its inputs are hidden. We could commit to the n^2 individual mask bits and check that they are bits, or apply (9) separately to all n^2 coordinates. Both alternatives are expensive. Instead, we combine the trace expressions before encoding the batched transposed mask. We commit in advance to auxiliary random masks and their squares; after the challenge, affine relations on these committed hidden values determine the batched transposed mask. This requires only $n - 1$ quadratic constraints.

4.1 Masking the message

Before sending the message, the prover samples independent uniform pads $p[0], \ldots, p[n-1] \in L$ and auxiliary masks $\rho[1], \ldots, \rho[n-1] \in L$. It computes $Q[j] = \rho[j]^2$ and commits to p, ρ, and Q together. The auxiliary masks will hide the challenge-dependent values used to certify the batched transposed mask.

We mask (3). The prover sends masked values

$$\widehat{m} = \widehat{s} + p. \tag{10}$$

The incoming claim y is carried as the public value $y' = y + p_{\text{in}}$, where p_{in} is a previously committed hidden pad. The first check (4) becomes the affine relation

$$\sum_v q_{\text{lo}}[v]p[v] + p_{\text{in}} = \sum_v q_{\text{lo}}[v]\widehat{m}[v] + y'. \tag{11}$$

Substituting $\widehat{m} = \widehat{s} + p$ and $y' = y + p_{\text{in}}$ cancels the pads and recovers (4). We include this affine relation in the final zero-knowledge proof.

The verifier now samples α as in Section 2. First, compute the left-hand side of (7) with $\widehat{m}$ in place of $\widehat{s}$, using the transpose formula (5). Call this the *batched masked value* m:

$$m = \sum_u \mathsf{eq}(u, \alpha) \sum_v \beta_v \widehat{m}[v][u]. \tag{12}$$

The bit-extraction identity (9), $x[u] = \mathrm{Tr}(\gamma_u x)$, writes the same value as

$$m = \sum_u \mathsf{eq}(u, \alpha) \sum_v \beta_v \,\mathrm{Tr}(\gamma_u \widehat{m}[v]).$$

Substituting (10) shows that the contribution of the pads is the *batched transposed mask*

$$\Delta = \sum_u \mathsf{eq}(u, \alpha) \sum_v \beta_v \,\mathrm{Tr}(\gamma_u p[v]). \tag{13}$$

Here the inner sum packs row u of the pad matrix, because $\mathrm{Tr}(\gamma_u p[v]) = p[v][u]$; the outer sum batches these transposed pad words with the same weights as (7). Additivity of the trace therefore gives

$$m = \sum_u \mathsf{eq}(u, \alpha) t[u] + \Delta. \tag{14}$$

Since $\Delta + \Delta = 0$, adding Δ recovers the unmasked left-hand side of (7). We must express this batched transposed mask as an affine function of the initial committed hidden values and public values.

The initial commitment and $\widehat{m}$ are fixed before α. If the challenge is derived from a transcript, that transcript includes both.

4.2 Combining the trace expressions

As written, the batched transposed mask (13) is a sum of n^2 terms, one for each bit of each pad, and certifying a bit of a hidden value costs a quadratic constraint. To certify the mask without reading these bits, we rewrite it as a sum of n powers of linear expressions in the pads. The rewriting groups the terms of the mask by pad, writes the resulting bit extraction as a sum of powers, and collects equal powers across the pads.

In the expression for Δ in (13), regrouping the terms by pad $p[v]$ gives

$$\Delta = \sum_{v=0}^{n-1} \beta_v \left(\sum_{u=0}^{n-1} \mathsf{eq}(u, \alpha) \,\mathrm{Tr}(\gamma_u p[v]) \right).$$

For fixed α, we abbreviate the inner weighted trace sum by defining

$$\varphi(x) = \sum_{u=0}^{n-1} \mathsf{eq}(u, \alpha) \operatorname{Tr}(\gamma_u x). \tag{15}$$

Thus $\varphi(p[v])$ is precisely the inner sum contributed by pad $p[v]$ to Δ. By the bit-extraction identity (9), $\varphi(x)$ is the sum of the bits $x[u]$ with the weights $\mathsf{eq}(u, \alpha)$. With this notation, the regrouped expression for Δ becomes

$$\Delta = \sum_{v=0}^{n-1} \beta_v \varphi(p[v]). \tag{16}$$

The trace is additive (Section 3), so φ is additive as well.

Expanding the trace in the definition of φ in (15), using $\operatorname{Tr}(z) = \sum_{j=0}^{n-1} z^{2^j}$ and $(\gamma_u x)^{2^j} = \gamma_u^{2^j} x^{2^j}$ from Section 3, and collecting equal powers of x gives the linearized polynomial [LN, Section 3.4]:

$$\varphi(x) = \sum_{j=0}^{n-1} \mu[j]\, x^{2^j}, \qquad \text{where} \qquad \mu[j] = \sum_{u=0}^{n-1} \mathsf{eq}(u, \alpha) \gamma_u^{2^j}. \tag{17}$$

The coefficients $\mu[j]$ depend only on the public challenge α and the fixed public basis, so both parties can compute them.

Writing the weighted bit extraction of each pad in (16) as the sum of powers (17) gives

$$\Delta = \sum_{j=0}^{n-1} \sum_{v=0}^{n-1} \mu[j] \beta_v\, p[v]^{2^j}.$$

Each public coefficient $\mu[j]\beta_v$ stands outside a power of a hidden pad. Equation (8) moves it inside: $\mu[j]\beta_v\, p[v]^{2^j} = \left((\mu[j]\beta_v)^{2^{n-j}} p[v]\right)^{2^j}$. For $0 \le j < n$, define the linear expression

$$T[j] = \sum_{v=0}^{n-1} (\mu[j]\beta_v)^{2^{n-j}}\, p[v]. \tag{18}$$

In characteristic two, the n terms with the same power 2^j add up to $T[j]^{2^j}$. Hence

$$\Delta = \sum_{j=0}^{n-1} T[j]^{2^j}. \tag{19}$$

Each $T[j]$ is a linear expression in the committed pads with public coefficients, so it needs no additional commitment and no quadratic constraint. The term $T[0]^{2^0} = T[0]$ is therefore affine. It remains to certify the $n-1$ powers $T[j]^{2^j}$ with $1 \le j < n$ without opening the commitment.

Soukhanov uses the linearized expression (17) for coordinate extraction in a proof system [S24, "Linearized formula for π"] and moves Frobenius powers onto public weights to batch openings at a Frobenius orbit of points [S24, "Improvement 1"]. Here we apply these techniques to the committed masking witnesses.

4.3 Certifying the batched transposed mask

We need an expression D, affine in committed and public values, with $D = \Delta$, so that the batched masked value m can be passed on with D as its mask. The powers $T[j]^{2^j}$ are not affine in the pads, and each squaring of a hidden value costs a quadratic constraint. Computing each power on its own takes j squarings, $n(n-1)/2$ in total. We evaluate the sum of these powers by a backwards recurrence using only $n-1$ squarings. The intermediate values depend on

the challenge α. To use the values committed before α (Section 5), the prover publishes these intermediate values under the auxiliary masks $\rho[j]$, committed in advance together with their squares $Q[j]$ (Section 4.1). Only the $n-1$ constraints $Q[j] = \rho[j]^2$ are quadratic.

After the challenge, the prover sets $X[n] = 0$ and computes backwards

$$X[j] = T[j] + X[j+1]^2, \qquad j = n-1, \ldots, 1. \tag{20}$$

Expanding this recurrence in characteristic two gives $X[1]^2 = \sum_{j=1}^{n-1} T[j]^{2^j}$, and the sum of powers (19) becomes

$$\Delta = T[0] + X[1]^2.$$

The prover publishes

$$Z[j] = X[j] + \rho[j], \qquad 1 \le j < n. \tag{21}$$

Each $Z[j]$ reveals nothing about $X[j]$, because its mask $\rho[j]$ hides exactly one published value. The recurrence (20) squares hidden intermediate values, and in characteristic two each such square splits into a public term and the committed square of its mask:

$$X[j]^2 = \big(Z[j] + \rho[j]\big)^2 = Z[j]^2 + Q[j].$$

Setting $Z[n] = \rho[n] = Q[n] = 0$ makes this split hold also for $j = n$, where $X[n] = 0$.

Substituting $X[j] = Z[j] + \rho[j]$ and these squares into (20) expresses every step of the recurrence through committed and public values. The zero-knowledge commitment scheme, for example Ligero [AHIV], verifies the input check (11), the quadratic constraints and the recurrence relations

$$Q[j] = \rho[j] \cdot \rho[j] \qquad (1 \le j < n), \tag{22}$$

$$T[j] = Z[j] + \rho[j] + Z[j+1]^2 + Q[j+1] \qquad (1 \le j < n). \tag{23}$$

The recurrence relations are affine, because $Z[j+1]^2$ is public and $T[j]$ is a linear expression in the pads by (18).

Since $\Delta = T[0] + X[1]^2$ and $X[1]^2 = Z[1]^2 + Q[1]$, the batched transposed mask equals the affine expression

$$D = T[0] + Z[1]^2 + Q[1]. \tag{24}$$

Lemma 1 proves $D = \Delta$ for every solution of the constraints above. For $j \in \{0,1\}^d$, let $a[j] = \varphi(q_{\mathrm{hi}}[j])$. By the bit-extraction identity (9) and the weighted bit extraction (15), $a[j]$ is the public coefficient of $\widehat{\mathsf{w}}[j]$ in the batched check (7). By (14), the unmasked left-hand side of that check equals $m + \Delta = m + D$. We pass on the *masked dot-product claim*

$$\langle a, \widehat{\mathsf{w}} \rangle = m + D. \tag{25}$$

Here m and a are public, and D stays hidden. Ring switching thus ends with a dot-product claim, as in Section 2, and the prover's messages $\widehat{m}$ and Z are masked. The dot-product claim is handled by the underlying commitment scheme, using D as its mask.

Cost. In total, the construction uses $n-1$ quadratic constraints, $n-1$ affine recurrence relations and one affine input check. The published values Z add $n-1$ field elements to the transcript. Ligero keeps the two inputs and the output of a product in separate slots, so the quadratic constraints occupy $3(n-1)$ slots and need $n-1$ more affine relations that tie the two equal inputs. At $n = 128$ this is 381 slots, not counting the pads p.

Computing the public coefficients and evaluating the linear expressions $T[j]$ still take $O(n^2)$ field operations, with the powers $\gamma_u^{2^j}$ and $\beta_v^{2^{n-j}}$ precomputed, because they depend only on the basis, which is fixed once and for all. The argument for the dot-product claim has its own cost. The saving concerns the hidden quadratic constraints: at $n = 128$ the construction needs 127 of them, against 8 128 for computing each power on its own and 16 384 for certifying the bits of the pads.

5 Correctness and security

We first prove completeness and soundness of our construction. Lemma 1 shows that, when the committed masks and their squares are consistent, the recurrence relations have a unique solution and this solution gives $D = \Delta$. Existence of the solution establishes completeness of the mask computation; the identity $D = \Delta$ for every satisfying solution establishes its soundness. Theorem 2 then shows that the new input check and masked dot-product claim are equivalent to the corresponding checks in the original ring-switching protocol. We use these results to establish completeness and soundness of the full reduction, and then give a zero-knowledge simulation under the assumptions on the underlying commitment scheme.

Lemma 1. *Fix α and the committed values p, ρ and Q, and suppose that the quadratic constraints (22) hold. Then the recurrence relations (23) have exactly one solution $Z[1], \dots, Z[n-1]$. For this solution, the affine expression (24) equals the batched transposed mask: $D = \Delta$.*

Proof. Use the terminal values $Z[n] = \rho[n] = Q[n] = 0$ and put $X[j] = Z[j] + \rho[j]$ for $1 \le j \le n$, so $X[n] = 0$. In characteristic two, the quadratic constraints give

$$X[j]^2 = (Z[j] + \rho[j])^2 = Z[j]^2 + Q[j].$$

Thus the recurrence relations (23) are equivalent to

$$T[j] = X[j] + X[j+1]^2, \qquad 1 \le j < n.$$

To construct a solution, start from $X[n] = 0$, compute $X[j] = T[j] + X[j+1]^2$ for $j = n-1, \dots, 1$, and set $Z[j] = X[j] + \rho[j]$. Substitution shows that these values satisfy every recurrence relation. Conversely, any satisfying solution must obey this same backwards recurrence, which fixes every $X[j]$ and hence every $Z[j]$. The solution therefore exists and is unique.

For this solution, write (20) as $T[j] = X[j] + X[j+1]^2$, raise it to the power 2^j and sum over $1 \le j < n$. In characteristic two,

$$\sum_{j=1}^{n-1} T[j]^{2^j} = \sum_{j=1}^{n-1} X[j]^{2^j} + \sum_{j=1}^{n-1} X[j+1]^{2^{j+1}} = X[1]^2 + X[n]^{2^n} = X[1]^2,$$

because every other term occurs twice and cancels. Substituting this into (19) gives

$$\Delta = T[0] + X[1]^2 = T[0] + Z[1]^2 + Q[1] = D. \qquad \square$$

Theorem 2. *Let $\widehat{s} = \widehat{m} + p$ be the hidden message, with rows $t[u]$ given by (5), and suppose that the quadratic constraints (22) and the recurrence relations (23) hold. Then the input check (11) is equivalent to the first check (4), and the masked dot-product claim (25) is equivalent to the batched check (7).*

Proof. Recall that $\widehat{s} = \widehat{m} + p$ and $y' = y + p_{\mathrm{in}}$. In characteristic two, the input check (11) is equivalent to

$$\sum_v q_{\mathrm{lo}}[v](\widehat{m}[v] + p[v]) = y' + p_{\mathrm{in}}.$$

Substituting the two identities gives $\sum_v q_{\mathrm{lo}}[v]\widehat{s}[v] = y$, which is the first check (4). Each step is reversible.

For the second equivalence, Lemma 1 gives $D = \Delta$. The mask-additivity identity (14) therefore gives

$$m + D = \sum_u \mathsf{eq}(u, \alpha) t[u] + \Delta + D = \sum_u \mathsf{eq}(u, \alpha) t[u].$$

By the definition $a[j] = \varphi(q_{\mathrm{hi}}[j])$ and the bit-extraction identity $\mathrm{Tr}(\gamma_u x) = x[u]$, we also have

$$a[j] = \sum_u \mathsf{eq}(u, \alpha) q_{\mathrm{hi}}[j][u].$$

Consequently, the masked dot-product claim (25) is exactly

$$\sum_{j \in \{0,1\}^d} \left(\sum_u \mathsf{eq}(u, \alpha) q_{\mathrm{hi}}[j][u] \right) \widehat{\mathsf{w}}[j] = \sum_u \mathsf{eq}(u, \alpha) t[u],$$

which is the batched check (7). □

Completeness. Suppose that the prover follows the protocol, so that $\widehat{s}$ is the honest message (3), and that the claim (2) is true. Substituting the honest message into the first check (4) gives the claim, so the first check holds. The batched check (7) holds because the row relation (6) holds for every u. The prover computes the intermediate values X and publishes Z as in Section 4.3. The quadratic constraints hold by the choice of Q, and the recurrence relations hold by construction. By Theorem 2, the input check and the masked dot-product claim hold as well. This holds for every challenge α and every choice of the prover's masks, so the reduction is perfectly complete. The resulting proof inherits the completeness of the underlying commitment scheme.

Soundness. Assume that the underlying commitment scheme is binding and sound, so accepted proofs certify the stated relations for fixed committed values. We show that the reduction has soundness error at most $\kappa/|L|$, where $\kappa = \log_2 n$ is the number of coordinates of the batching challenge α. The commitments and the masked message $\widehat{m}$ are fixed before α, so $\widehat{s} = \widehat{m} + p$ and its transposed rows $t[u]$ are fixed as well.

Suppose that the incoming claim (2) is false and that the input check holds. By Theorem 2, $\widehat{s}$ satisfies the first check (4), so it cannot be the honest message (3). Transposition is invertible, so the row relation (6) implies that the row errors

$$e[u] = t[u] - \sum_{j \in \{0,1\}^d} q_{\mathrm{hi}}[j][u] \widehat{\mathsf{w}}[j]$$

are not all zero.

The batched check (7) is equivalent to $E(\alpha) = 0$, where

$$E(\alpha) = \sum_u e[u] \mathsf{eq}(u, \alpha).$$

Each weight $\mathsf{eq}(u, \alpha)$ is a product of κ affine factors, so E has total degree at most κ. At each Boolean point u, it takes the value $e[u]$; since the row errors are not all zero, E is a nonzero polynomial. Its coefficients are fixed before α, so the Schwartz–Zippel lemma gives

$$\Pr_{\alpha \leftarrow L^\kappa}[E(\alpha) = 0] \le \frac{\kappa}{|L|}.$$

This is the batching argument of [DP24, Theorem 3.5].

Lemma 1 forces $D = \Delta$ for every satisfying choice of Z, including choices made after α. By Theorem 2, the masked dot-product claim therefore holds exactly when $E(\alpha) = 0$. Thus a false incoming claim can pass the input check and produce a true outgoing claim with probability at most $\kappa/|L|$, which is $7/2^{128}$ for $n = 128$.

Zero knowledge. We sketch a simulator for perfect zero knowledge in the interactive oracle proof (IOP) model against malicious verifiers within the underlying IOP's query bound: given only a true public input, it exactly reproduces the verifier's view. We assume perfectly hiding commitments and a perfect joint simulator for the oracle answers, constraint checks and outgoing claim that respects the commitments made before α and handles adaptive queries.

Given the public input, the simulator runs the malicious verifier and starts the underlying simulation for the oracle interaction preceding the challenge. It samples $\widehat{m} \leftarrow L^n$ uniformly and sends it to the verifier, passing its oracle queries to the same underlying simulator throughout. When the verifier supplies α, the simulator uses that actual challenge and samples $Z \leftarrow L^{n-1}$ uniformly and independently. It computes m and a from these public values as in Section 4.1 and Section 4.3. Using the same underlying simulation state, it then completes the oracle answers and proofs for the constraints and the outgoing claim. It never needs the witness or the actual pads.

The explicit messages have exactly the required distribution. For any honest message $\widehat{s}$, the fresh uniform pads make $\widehat{m} = \widehat{s} + p$ uniform, also given the incoming masked value y'. Perfect hiding of the preceding oracle interaction leaves the auxiliary masks $\rho[j]$ uniform, even conditioned on the verifier's view before it chooses α. For fixed p and that challenge, the recurrence fixes X, so the values $Z[j] = X[j] + \rho[j]$ are jointly uniform. This argument allows α to depend on the earlier messages and oracle answers. The simulator preserves that dependence by running the verifier, and the joint simulation preserves the correlations with subsequent oracle answers. The public values m and a are deterministic functions of this transcript, so their distributions agree as well.

For any sampled $\widehat{m}$ and Z, suitable hidden values exist: take $p = \widehat{m} + \widehat{s}$ for the honest message, compute X from these pads and α, and put $\rho[j] = Z[j] + X[j]$ and $Q[j] = \rho[j]^2$. Completeness shows that these values satisfy the constraints and the outgoing claim. The assumed joint simulator supplies the remaining oracle view consistently with the earlier interaction, giving perfect zero knowledge against the stated class of malicious verifiers for the composed IOP. With a cryptographic commitment instantiation, the resulting zero-knowledge guarantee is that of the underlying simulator; perfect simulation of the explicit messages alone does not make that guarantee perfect.

Acknowledgments

The author thanks Mark Van de Vyver for helpful comments on an earlier version of this note.

References

[AHIV] S. Ames, C. Hazay, Y. Ishai, and M. Venkitasubramaniam. Ligero: Lightweight sublinear arguments without a trusted setup. Cryptology ePrint Archive, Paper 2022/1608, 2022. https://eprint.iacr.org/2022/1608.

[D25] B. E. Diamond. Zero-knowledge polynomial commitment in binary fields. Cryptology ePrint Archive, Paper 2025/1015, 2025. https://eprint.iacr.org/2025/1015.

[DP24] B. E. Diamond and J. Posen. Polylogarithmic proofs for multilinears over binary towers. Cryptology ePrint Archive, Paper 2024/504, 2024. https://eprint.iacr.org/2024/504. Numbering follows the version of May 14, 2026.

[LN] R. Lidl and H. Niederreiter. *Finite Fields*. Cambridge University Press, second edition, 1997.

[S24] L. Soukhanov. Hashcaster. HackMD note, 2024. https://hackmd.io/@levs57/SJ4fuZMD0.

[Binius] The Binius developers. Binius64 Blueprint. Protocol specification, October 7, 2026. https://www.binius.xyz/spec.pdf.