Lumora: A Family of Permutation-Based Wide-Block Ciphers for Post-Quantum zkSNARK Applications

Susanta Samanta[1], Martin Grenouilloux[2], Guang Gong[1], and Chunlei Li[2]

[1] Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada
{ssamanta,ggong}@uwaterloo.ca
[2] Department of Informatics, University of Bergen, Bergen, N-5020, Norway
{martin.grenouilloux,chunlei.li}@uib.no

Abstract. The deployment of advanced cryptographic protocols such as zero-knowledge proofs (ZKPs) requires symmetric primitives optimized for fast verification inside proof systems. In frameworks based on Rank-1 Constraint Systems (R1CS), prover performance and proof size are dominated by the cost of arithmetization, specifically, by the number of nonlinear multiplication constraints. Traditional bit-oriented designs are typically inefficient under this metric. In this paper, we introduce Lumora, a family of arithmetization-oriented, permutation-based wide-block ciphers designed for efficient use inside zkSNARK circuits and for applications in post-quantum digital signatures. Each instance of Lumora follows a unified AES-like SPN structure defined over the binary extension field $\mathbb{F}_{2^n}$ for $n \in \{16, 32, 64\}$. The underlying permutation is instantiated as a block cipher via the Even–Mansour paradigm, which eliminates the R1CS constraint overhead of a separate key schedule, ensuring the prover's workload remains strictly focused on evaluating the public permutation. Finally, we provide a detailed security analysis of the Lumora family, together with implementation results and a comparison within the FAEST-EM-256 framework.

Keywords: zkSNARK · R1CS · arithmetization-oriented primitive · binary extension field · SPN · MDS matrix · FAEST

1 Introduction

Privacy-focused protocols like zero-knowledge proofs (ZKPs), secure multiparty computation (MPC), and fully homomorphic encryption (FHE) have recently pushed symmetric cryptography toward a new goal, not only efficient execution on conventional hardware, but also efficient verification inside proof systems. Widely deployed designs such as AES and SHA-2/SHA-3 are primarily optimized for standard architectures, where bitwise operations are extremely cheap.

However, when such primitives are used inside a ZKP system, performance is often dominated by arithmetization cost, that is, the cost of expressing the computation as polynomial constraints over a finite field. In this setting, bit-oriented designs can be inefficient because their operations do not map naturally to field arithmetic, often leading to more constraints and higher proving cost.

This has motivated the development of new arithmetization-oriented (AO) symmetric primitives. These designs aim to minimize expensive constraints, especially nonlinear ones such as field multiplications, while still providing strong security. Early proposals such as MiMC [2] helped establish this direction, and the design space has evolved rapidly since then. POSEIDON [36] is a well-known hash function designed for efficient use inside ZKP systems and has seen broad adoption in verifiable-computation and related applications. POSEIDON2 [37] further improves performance, particularly over smaller prime fields, while recent work also considers POSEIDON(2)b [33] as variants over binary extension fields targeting 128-bit security.

Beyond MiMC and POSEIDON, many other AO primitives have been proposed, including MARVELlous family [7], Rescue [5], Anemoi [17], Griffin [32], RAIN [26]. Within the AO landscape, several designs further exploit lookup arguments to reduce constraint costs in lookup-enabled proving systems, including Reinforced Concrete [34], Monolith [35] and Polocolo [42]. For a broader overview of such designs, together with references to known cryptanalysis and implementation results, we refer the reader to the STAP Zoo resource [52], which curates symmetric primitives for advanced protocols and tracks subsequent security developments.

ZKP frameworks typically express computations using arithmetization paradigms such as Rank-1 Constraint Systems (R1CS), Plonk style arithmetizations, and Algebraic Intermediate Representation (AIR). In particular, R1CS underlies popular SNARKs such as Groth16 [39] and is also used in systems such as Aurora [14]. An R1CS instance represents computations as a sequence of matrix equations of the form $(A \cdot z) \times (B \cdot z) = C \cdot z$, where z is the witness vector and A, B, C are constraint matrices. In R1CS, linear operations such as additions are essentially free, as they can be absorbed into the linear combinations, whereas each multiplication requires a dedicated constraint. Consequently, in R1CS-based zkSNARKs, efficiency is driven to a large extent by the number of multiplication constraints, making their reduction a central design objective.

Motivated by this cost model, we introduce in this paper Lumora, a family of permutation-based wide-block ciphers designed for efficient use inside zkSNARK circuits and for applications in post-quantum digital signatures [11,12,13]. Each instance of Lumora follows a unified AES-like SPN structure, with the block size as the main varying parameter. More specifically, Lumora$(16n, n)$ is defined over the binary extension field $\mathbb{F}_{2^n}$ for $n \in \{16, 32, 64\}$, where $16n$ denotes the block size in bits. We instantiate the underlying public permutation as a block cipher using a simple keying mechanism such as the Even–Mansour paradigm [30], which keeps the proving circuit focused on permutation evaluation and avoids additional constraints from a separate key schedule.

We evaluate the Lumora family against a broad spectrum of statistical and algebraic attacks and use the resulting bounds to determine the round count of each instance. To thwart classical statistical threats, Lumora strictly adheres to the wide trail design strategy [21]. It guarantees a minimum of 25 active Sboxes over any four-round differential or linear trail. With the optimal differential uniformity and nonlinearity of the inverse mapping, the maximum differential probability and absolute linear correlation of Lumora drop below the required security thresholds after just four rounds. Furthermore, combining this low differential probability with full diffusion within two rounds drastically reduces the likelihood of constructing short distinguisher trails, securely thwarting impossible differential and boomerang attacks. Potential structural vulnerabilities, such as invariant subfield attacks, are explicitly neutralized by injecting asymmetric round constants and the selection of the coefficients of the affine polynomial. Lumora mitigates the vulnerabilities associated with partial rounds by applying its inverse Sbox uniformly within a full SPN architecture. Coupled with this uniform structure, the maximum density of the affine polynomial rapidly increases the algebraic complexity of the cipher, effectively thwarting advanced algebraic cryptanalysis.

We also present an `x86-64` optimized reference implementation in `C` using CLMUL instructions. Finally, within the FAEST-EM-256 framework, Lumora$(256, 16)$ reduces the witness length from 2688 bits to 2304 bits and decreases the projected signature sizes for the small and fast parameter sets from 17,984 to 16,928 bytes and from 23,476 to 21,940 bytes, respectively. Table 4 summarizes these comparative results.

2 Comparison with Related Work

We briefly position Lumora among prior AO symmetric primitives, with particular emphasis on designs over binary extension fields. Rather than aiming at an exhaustive survey, we highlight the main design tradeoffs between Lumora and representative prior approaches, especially with respect to algebraic structure, nonlinear layer design, and R1CS efficiency.

One of the earliest proposals in this direction is MiMC (Minimal Multiplicative Complexity) [2]. MiMC is highly flexible and can be instantiated over binary fields $\mathbb{F}_{2^n}$ using the simple power map $x \mapsto x^3$. Its elegant design also motivated later variants such as GMiMC [4] and the POSEIDON family [36]. However, its reliance on the cubic map makes it structurally vulnerable to algebraic cryptanalysis. Although the original MiMC design selected its parameters with interpolation and GCD-based attacks in mind, subsequent research still yielded algebraic attacks against the full cipher [28]. Lumora employs the multiplicative inverse, achieving the maximum algebraic degree for a permutation over $\mathbb{F}_{2^n}$.

JARVIS [7] was one of the earliest STARK-friendly symmetric designs operating over binary extension fields. It applies a single inverse Sbox over the whole state, followed by an affine transformation. However, it was later broken by Gröbner-basis attacks that exploited a structural weakness in this affine

layer [3]. More precisely, the affine polynomial $A(x)$ in JARVIS admits a decomposition $A(x) = C \circ B^{-1}(x)$, where $B(x)$ and $C(x)$ are quartic polynomials. This decomposition allows consecutive rounds to be described using simpler equations, without explicitly expanding the high-degree polynomial $B^{-1}(x)$ explicitly [3].

To mitigate this vulnerability, subsequent designs explored two divergent paths. RAIN [26] preserves the single-large-Sbox design over the full state, pairing it with an unstructured random affine layer over $\mathbb{F}_2$ to eliminate the algebraic regularity needed for decomposition attacks. However, this unstructured approach remains vulnerable to advanced algebraic techniques. Specifically, Liu et al. [49] demonstrated that RAIN admits equivalent algebraic representations that can be exploited to break the 2-round variant. More recently, Yang et al. [57] developed a polynomial decomposition method over the extension field $\mathbb{F}_{2^n}$ that successfully attacks both the 2-round and full 3-round versions of RAIN.

Vision [5] moves away from the single-large-Sbox setting and adopts an SPN structure that applies smaller Sboxes in parallel across state elements. This comes at the expense of a heavier, two-step round architecture to maintain security, which significantly increases the total number of R1CS constraints. Lumora retains the inverse-based philosophy, but replaces single-large-Sbox design with a 16-cell AES-like SPN. Moreover, its affine map is induced by the block matrix M_L (see (1)), and for the instantiated parameters the associated linearized polynomial $L(x) = \sum_{i=0}^{n-1} a_i x^{2^i}$ has all coefficients $a_i \neq 0$, which suggests a structurally denser linearized representation than the one exploited in JARVIS.

Anemoi [17], Griffin [32], and the recently introduced POSEIDON(2)b [33] represent another branch of AO primitives. They employ low-degree power maps, such as $x \mapsto x^3$ or $x \mapsto x^5$, to minimize the nonlinear constraints. However, a power map defines a permutation over $\mathbb{F}_{2^n}$ only if its exponent is coprime to $2^n - 1$. For hardware-aligned even extension degrees (e.g., $n \in \{16, 32, 64\}$), the exponent 3 is never coprime to $2^n - 1$, and 5 fails when $n \equiv 0 \pmod 4$. Consequently, POSEIDON(2)b employ larger valid exponents, such as x^7, which increases the R1CS constraint cost per Sbox. To compensate for this overhead, POSEIDON(2)b relies on HADES design strategy [38]. Yet, very recent cryptanalysis [53] reveals that combining partial rounds with highly structured matrices creates invariant subspaces. The authors exploited these vulnerabilities to improve round-skipping for the constrained-input constrained-output (CICO) problem on the underlying permutation.

Lumora is designed to avoid these mathematical limitations and structural tradeoffs discussed above. By employing the multiplicative inverse x^{-1}, it provides each cell maximal algebraic degree over the base field $\mathbb{F}_{2^n}$ while maintaining the minimum R1CS cost per Sbox. Moreover, Lumora mitigates the vulnerabilities associated with partial rounds by applying its inverse Sbox to all 16 state cells in every round. Although Lumora incorporates a tensor product matrix (M_L), this structure is strictly confined to the Sbox to define the affine polynomial. Global state diffusion is instead provided by MDS-based MixColumns together with ShiftRows in every round. Because every cell undergoes both a high-degree inversion and optimal MDS mixing during each step, the subspace

trails and round-skipping techniques that compromise POSEIDON(2)b are expected to be rendered structurally infeasible.

Alternative Modes and Large-State Traditional Primitives. It is also worth noting that the Lumora permutation can be deployed directly in a sponge construction. Moreover, Lumora could serve as the round or update permutation in a Feistel or a nonlinear feedback shift register (NLFSR) built from two $16n$-bit branches, yielding a $32n$-bit internal state. For example, choosing $n = 32$ gives a 1024-bit state, in that case, roughly twice as many rounds would typically be needed to maintain desired security.

Adapting traditional large-state symmetric primitives to the AO setting remains an open challenge. For example, Keccak-1600 [15] has a 1600-bit internal state and is highly efficient on platforms supporting parallel 64-bit operations, but its round function is not naturally expressed as a low multiplicative complexity permutation over a large binary extension field such as $\mathbb{F}_{2^{64}}$. Similarly, the SNOW-V [29] stream cipher maintains a large 896-bit internal state, utilizing two 16-stage LFSRs over $\mathbb{F}_{2^{16}}$ with a Finite State Machine (FSM) comprising three 128-bit registers. To update the FSM, SNOW-V employs instances of the AES encryption round function. Because its nonlinear permutation operates over the much smaller $\mathbb{F}_{2^8}$ field (via the AES Sbox). It therefore remains unclear how such architectures could be reformulated to meet the low multiplicative complexity requirements that motivate AO design. In contrast, Lumora$(16n, n)$ is designed from the outset to satisfy this objective.

3 Design Specifications

The block ciphers adopt the single-key Even–Mansour construction [30], defined as $E_k(P) = k \oplus \text{Lumora}(P \oplus k)$, where the Lumora family, denoted by Lumora$(16n, n)$, follows a unified SPN structure obtained by iterating round functions. The instances Lumora(256, 16), Lumora(512, 32), and Lumora(1024, 64) iterate 12, 10, and 8 rounds, respectively.

Fix $n \in \{16, 32, 64\}$ and let $\mathbb{F}_{2^n}$ be the finite field used by the corresponding instance. A Lumora instance operates on a $16n$-bit state, viewed as a 4×4 array state $\mathbb{S} = (s_{i,j})$ of sixteen n-bit cells, where $s_{i,j} = s_{i+4j} \in \mathbb{F}_{2^n}$ for $0 \le i, j < 4$. Figure 1 visually represents the round function of Lumora.

3.1 The SubCell Transformation η

The SubCell transformation η is a nonlinear transformation in which n-bit Sbox S is applied to every cell of the cipher internal state

$$s_j \leftarrow S(s_j) \quad \text{for } j = 0, 1, \ldots, 15.$$

The Sbox S is defined as $S(x) = L(y) + a_n$, where $y = x^{-1}$ is the multiplicative inverse of x in $\mathbb{F}_{2^n}$ (with $y = 0$ when $x = 0$), and L is a linearized permutation polynomial over the finite field $\mathbb{F}_{2^n}$. The vector a_n is a fixed nonzero element in $\mathbb{F}_{2^n}$, and its choice will be specified later.

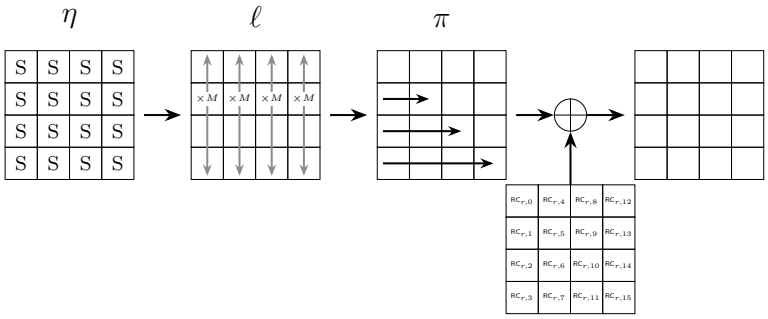


Fig. 1. The round function applies three different transformations: η, ℓ, π, followed by the operation of adding round constants.

3.1.1 Unified Structure of the Linearized Polynomial L: Similarly to the AES Sbox, we construct the Sbox for Lumora by composing the multiplicative inverse operation with an affine polynomial. Recall that a linearized polynomial over $\mathbb{F}_{2^n}$ takes the form $L(x) = \sum_{i=0}^{n-1} a_i x^{2^i} \in \mathbb{F}_{2^n}[x]$. Such a polynomial is a permutation on $\mathbb{F}_{2^n}$ if and only if the associated Dickson matrix, whose first row is $\{a_0, a_1, \ldots, a_{n-1}\}$,

$$\begin{bmatrix} a_0 & a_1 & \cdots & a_{n-1} \\ a_{n-1}^2 & a_0^2 & \cdots & a_{n-2}^2 \\ \vdots & \vdots & & \vdots \\ a_1^{2^{n-1}} & a_2^{2^{n-1}} & \cdots & a_0^{2^{n-1}} \end{bmatrix}$$

is nonsingular [48]. By adding a constant term to this linearized polynomial, we obtain an affine polynomial of the form

$$A(x) = \sum_{i=0}^{n-1} a_i x^{2^i} + a_n \in \mathbb{F}_{2^n}[x].$$

To present the linearized polynomial used in the Sbox uniformly across all parameter sets, we begin by introducing the following block matrix:

$$M_L = \begin{bmatrix} 0_m & 0_m & I_m & I_m \\ I_m & 0_m & 0_m & 0_m \\ I_m & I_m & 0_m & 0_m \\ 0_m & 0_m & I_m & 0_m \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \otimes I_m \tag{1}$$

where 0_m denotes the $m \times m$ zero matrix, and I_m is the $m \times m$ identity matrix, with $m = \frac{n}{4}$. Therefore, M_L is an $n \times n$ binary matrix. This matrix provides a compact representation of the linearized polynomial common to all instances of the Lumora Sbox.

The polynomial L over $\mathbb{F}_{2^n}$ is then defined as the linearized polynomial associated with the binary matrix M_L, taken with respect to the standard basis $\{1, \alpha, \alpha^2, \ldots, \alpha^{n-1}\}$, where α is a primitive element in $\mathbb{F}_{2^n}$. We now recall the following result.

Theorem 1. *[23] Let $\{e_1, e_2, \ldots, e_n\}$ be a basis of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$ with dual basis $\{d_1, d_2, \ldots, d_n\}$, and let M be an $n \times n$ matrix over $\mathbb{F}_2$. Then the linear transformation induced by M can be represented by the linearized polynomial $L(x) = \sum_{i=0}^{n-1} a_i x^{2^i}$ with the coefficients a_i given by*

$$a_i = \sum_{j=1}^{n} \sum_{k=1}^{n} M_{jk} d_k^{2^i} e_j.$$

In the Lumora family, the underlying finite field is $\mathbb{F}_{2^n}$ with $n \in \{16, 32, 64\}$. According to Theorem 1, for the parameter choices (see Section 3.5), all coefficients a_i of the linearized polynomial L are nonzero. Moreover, the coefficients $\{a_0, a_1, \ldots, a_{n-1}\}$ guarantee that the associated Dickson matrix is nonsingular. Consequently, the linearized polynomial L acts as a permutation over $\mathbb{F}_{2^n}$ when n is 16, 32, or 64. Additionally, none of the coefficients of L coincide with α. Therefore, the coefficient a_n is set to α, which ensures that the resulting affine mapping facilitates an efficient and uniform description of the Sbox across all Lumora variants.

3.2 The MixColumns Transformation ℓ

The MixColumns transformation ℓ is a linear operation that acts independently on each of the four columns of the state. It is defined using a 4×4 MDS matrix M over $\mathbb{F}_{2^n}$. For each column, we apply

$$(s_j, s_{j+1}, s_{j+2}, s_{j+3})^t \leftarrow M \cdot (s_j, s_{j+1}, s_{j+2}, s_{j+3})^t \text{ for } j = 0, 4, 8, 12.$$

3.2.1 Unified Structure of the Matrix M: To describe the diffusion layer in a uniform way across all instances of Lumora, we specify the MixColumns matrix M through a common algebraic structure that is independent of the choice of n.

The MDS matrix M is specified uniformly for all parameter sets as the product of four sparse 4×4 matrices M_1, M_2, M_3, and M_4, i.e., $M = M_1 M_2 M_3 M_4$, where

$$M_1 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, M_2 = \begin{bmatrix} 0 & 0 & 1 & \alpha \\ 1 & 0 & 0 & 0 \\ \alpha^{-1} & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, M_3 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ \alpha^{-1} & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \text{ and } M_4 = M_1.$$

The resulting matrix M is then

$$M = \begin{bmatrix} \alpha^{-1}+1 & \alpha^{-1} & 1 & \alpha^{-1}+1 \\ \alpha+1 & \alpha & \alpha^{-1} & \alpha^{-1} \\ \alpha & \alpha+1 & \alpha^{-1}+1 & \alpha^{-1} \\ \alpha^{-1} & \alpha^{-1} & \alpha^{-1}+1 & 1 \end{bmatrix}.$$

For implementation, the MixColumns operation is realized by applying the sparse matrices sequentially, namely $M \cdot \hat{y} = M_1 \cdot \big(M_2 \cdot \big(M_3 \cdot (M_4 \cdot \hat{y})\big)\big)$, for any column vector $\hat{y} \in \mathbb{F}_{2^n}^4$. This factorization reduces the cost of the linear layer, since only three multiplications in $\mathbb{F}_{2^n}$ are required and all remaining operations are additions.

3.3 The ShiftRows Operation π

The ShiftRows operation π cyclically rotates row i of the state array by i cell positions to the right, for $i = 0, 1, 2, 3$. In terms of the linear indexing of the state, this is expressed as $s_j \leftarrow s_{13 \cdot j \bmod 16} \quad \text{for } j = 0, 1, \ldots, 15$.

3.4 Round Constants

Iterated ciphers using identical round functions naturally exhibit structural symmetry. Lumora breaks these symmetries by XORing each state word s_j in round r with a distinct constant $\mathsf{RC}_{r,j}$. The constants are deterministically generated from the decimal expansion of π, beginning with the leading digit in $\pi = 3.14159\ldots$, by greedily consuming consecutive digits until an n-bit value is obtained. This extraction proceeds continuously across rounds, so each round uses the next unused digits of π. For the parameter sets Lumora$(256, 16)$, Lumora$(512, 32)$, and Lumora$(1024, 64)$, this yields 192, 160, and 128 constants $\mathsf{RC}_{r,j}$, respectively.

3.5 The Parameters

The Lumora family discussed here includes three instances, denoted Lumora$(16n, n)$ for $n \in \{16, 32, 64\}$, defined over $\mathbb{F}_{2^n}$ using the primitive polynomials in Table 1, where R1CS for each Sbox requires only one multiplication. For each instance, the block matrix M_L determines the linearized polynomial $L(x) = \sum_{i=0}^{n-1} a_i x^{2^i}$ with respect to the standard basis $\{1, \alpha, \alpha^2, \ldots, \alpha^{n-1}\}$, where α is a root of the chosen primitive polynomial. This $L(x)$ induces a permutation over $\mathbb{F}_{2^n}$ where all coefficients a_i are nonzero. Explicit values for these coefficients are listed in Appendix A.

Table 1. Lumora instantiation parameters.

Variant	Block Size	Field	Defining Polynomial	Sbox Degree	Rounds	R1CS
Lumora(256, 16)	256 bits	$\mathbb{F}_{2^{16}}$	$x^{16} + x^{12} + x^3 + x + 1$	15	12	192
Lumora(512, 32)	512 bits	$\mathbb{F}_{2^{32}}$	$x^{32} + x^{22} + x^2 + x + 1$	31	10	160
Lumora(1024, 64)	1024 bits	$\mathbb{F}_{2^{64}}$	$x^{64} + x^4 + x^3 + x + 1$	63	8	128

4 Design Rationale

In this section, we explain the design decisions behind the Lumora family. Lumora follows an AO design approach. It operates over the binary extension fields $\mathbb{F}_{2^n}$ for $n \in \{16, 32, 64\}$, employs the multiplicative inverse as its Sbox, and uses an AES-like SPN structure with a 4×4 MDS matrix to ensure optimal diffusion. Lumora is designed first and foremost as a wide, algebraically zkSNARK-friendly permutation; the Even–Mansour construction is then chosen as the simplest way to turn this permutation into a block cipher without changing that structure.

4.1 Even–Mansour Construction

Lumora adopts the single-key Even–Mansour (EM) construction [30], defined as $E_k(P) = k \oplus \text{Lumora}(P \oplus k)$, so the only keyed operations are two XORs with a $16n$-bit key, and the round function of Lumora itself remains completely keyless. From a security standpoint, the Even–Mansour construction has been extensively analyzed [20,27]. In the random b-bit permutation model, single-key EM provides security up to $\mathcal{O}(2^{b/2})$ queries when the adversary has access to both encryption and decryption oracles. Recent results on post-quantum security [1] have established that Even–Mansour maintains meaningful security against quantum adversaries, though with reduced classical margins.

From an implementation perspective, EM eliminates the need for a complex key schedule, significantly reducing implementation complexity. For ZKP applications (as used in FAEST [13]), the variant where the permutation is public and only the input is secret simplifies proof generation by avoiding key schedule simulation. This dramatically reduces the number of R1CS constraints to only those from the permutation itself, enabling efficient ZKP-based signatures. Overall, these properties make the single-key Even–Mansour paradigm a natural choice for instantiating Lumora as a block cipher, preserving its permutation-based structure while matching our security and zkSNARK-friendliness goals.

4.2 The Inverse Sbox

The choice of the Sbox is one of the critical design decisions for an AO cipher. In Lumora, the Sbox is constructed as a composition of the multiplicative inverse function over $\mathbb{F}_{2^n}$ and a linearized polynomial L, followed by a constant. Formally, it is defined as $S(X) = L(Y) + a_n$, where $Y = X^{-1}$ denotes the multiplicative inverse of X in $\mathbb{F}_{2^n}$ (with $Y = 0$ when $X = 0$). This choice is motivated by the following considerations.

R1CS Efficiency. Lumora minimizes R1CS constraints for zkSNARK efficiency. In the R1CS constraint system, the inverse function over $\mathbb{F}_{2^n}$ counts as a single constraint, despite having algebraic degree $n - 1$. This is because verifying $y = x^{-1}$ reduces to checking $x \cdot y = 1$ (with a special case for $x = 0$), which requires only one multiplication gate.

Cryptographic Properties. Based on Nyberg's analysis [54], the inverse Sbox for even n provides differential uniformity 4, bounding the maximum differential probability by 2^{-n+2}. Lumora instances ($n \in \{16, 32, 64\}$) leverage this for optimal differential property. The nonlinearity $2^{n-1} - 2^{n/2}$ results in a maximum absolute linear correlation of $2^{-n/2+1}$. For example, Lumora(256, 16) achieves a maximum differential probability of 2^{-14} and a maximum absolute linear correlation of 2^{-7}. Moreover, its maximal algebraic degree of $n-1$ provides robust resistance against higher-order differential and algebraic attacks [45,46].

The Linearized Polynomial. The linearized polynomial L over $\mathbb{F}_{2^n}$ is defined via the binary matrix M_L (see (1)) with a unified block structure across all field sizes. Computing $L(Y)$ is equivalent to the matrix product $M_L \cdot Y$ with $Y \in \mathbb{F}_2^n$. Since M_L is very sparse, this significantly reduces the computational cost. The block matrix M_L is chosen randomly from the class of GDLS matrices [40] so that, when viewed as an $n \times n$ binary matrix, it yields a linearized polynomial over $\mathbb{F}_{2^n}$ with respect to the standard basis $\{1, \alpha, \alpha^2, \ldots, \alpha^{n-1}\}$, as obtained from M_L via Theorem 1. Also, a linearized polynomial $L(x) = \sum_{i=0}^{n-1} a_i x^{2^i} \in \mathbb{F}_{2^n}[x]$ is a permutation on $\mathbb{F}_{2^n}$ if and only if its associated Dickson matrix with coefficients $\{a_0, a_1, \ldots, a_{n-1}\}$ as the first row, is nonsingular [48]. Lumora fulfills this requirement while explicitly selecting M_L such that all the coefficients a_i are strictly nonzero. This yields a dense linearized representation and contributes to the monomial growth analyzed in Section 5.1. Finally, the constant $a_n = \alpha$ is additionally chosen to eliminate the fixed point of the Sbox at zero.

4.3 Linear Layer

Lumora employs a 4×4 MDS matrix M over $\mathbb{F}_{2^n}$ for its MixColumns operation. While Near-MDS matrices or matrices with an even lower branch number often offer more efficient implementations, they reduce diffusion speed and necessitate more rounds than designs that employ MDS matrices. The diffusion speed is determined by the number of rounds required to achieve full diffusion. Here, full diffusion in a block cipher refers to the property that every state bit depends on all state bits from some rounds earlier. In Lumora, full diffusion is achieved before the end of the second round (see Figure 2). Although, MDS matrices

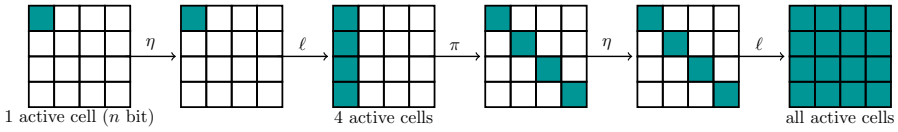


Fig. 2. Full diffusion in Lumora family.

are not sparse, they can be obtained from sparse matrices using recursive or

compositional constructions. In Lumora, the MDS matrix M is realized as the product of four sparse matrices as $M = M_1 \cdot M_2 \cdot M_3 \cdot M_4$.

These matrices M_i belong to the GDLS matrix family introduced in [40], and a similar GDLS-based construction is employed in the linear layer of the FUTURE [41]. The implementation cost of the MDS matrix M is significantly reduced as the cost of M is simply the sum of the implementation costs of M_1, M_2, M_3, and M_4. More specifically, we need only three multiplications in $\mathbb{F}_{2^n}$, and $8n$ bitwise XORs in total. Moreover, since the multiplications are by α and α^{-1}, their cost is among the lowest for nonzero elements in the field, excluding multiplication by one.

Recall that a $k \times k$ matrix A is MDS if the set $\{(x, xA) : \ x \in \mathbb{F}_{2^n}^k\}$ forms an MDS code, i.e., a linear code of length $2k$, dimension k, and minimum distance $k+1$. From the characterization of MDS codes [50, Page 321], this holds if and only if all square submatrices of A are nonsingular (equivalently, all minors are nonzero). Thus, it suffices to verify that every minor of A is nonzero.

A direct computation of the minors of M shows that the irreducible factors appearing in them form the set

$$\{\alpha, \alpha+1, \alpha^2+\alpha+1, \alpha^3+\alpha+1, \alpha^3+\alpha^2+1\}$$

Consequently, when α is chosen as a root of the defining polynomial $t(x)$, none of these factors vanishes, implying all minors of M are nonzero. In particular, M is an MDS matrix over $\mathbb{F}_{2^n}$. Combined with the ShiftRows, the MDS matrix ensures that, for all variants of Lumora, the minimum number of active Sboxes over four rounds is at least 25, following the wide trail design strategy [21].

4.4 The Number of Rounds

Two thresholds could in principle arise in the following analysis. We therefore fix the convention that all cryptanalysis targets the Lumora permutation as a standalone object, for which the relevant reference is the full-codebook threshold $2^{-b} = 2^{-16n}$. We analyze the permutation exclusively against this stronger reference. For completeness, we note that when the permutation is keyed through the Even–Mansour construction, the generic security of the mode is capped at $O(2^{b/2})$ queries, as recalled in Section 4.1. This bound is a property of the mode and is independent of the round count.

Consistently with this convention, the security analysis in Section 5 shows that the reach of every known cryptanalytic technique against Lumora is confined to a small number of rounds. On the statistical side, the wide-trail strategy guarantees at least 25 active Sboxes over any four rounds. Since the maximum differential probability of the inverse Sbox is 2^{-n+2}, the probability of any specified four-round differential trail is upper bounded by $(2^{-n+2})^{25} = 2^{-25(n-2)}$, which clears the threshold 2^{-16n} with a wide margin for all $n \in \{16, 32, 64\}$. The full diffusion within two rounds additionally closes the short distinguishers exploited by boomerang and impossible-differential attacks. On the algebraic side, the Gröbner-basis and FreeLunch analyses of Section 5.1 already exceed the 2^b

target for three or more rounds. The integral analysis of Section 5.3 reaches four rounds, at cost $O(2^{5n+2})$ with 2^n chosen states, and does not extend further. We therefore use four rounds as a conservative lower bound for the security analysis.

Because Lumora adopts an AES-like SPN layout, its permutation may admit structural attacks developed for AES-like designs, whose round coverage is largely independent of the field size n. Their concrete cost, however, is dominated by cell-wise operations over $\mathbb{F}_{2^n}$ and thus grows sharply with n relative to AES over $\mathbb{F}_{2^8}$, as illustrated by the $O(2^{5n+2})$ complexity of the four-round integral attack. We do not re-derive all such attacks for Lumora's wider state and larger field, and a dedicated analysis of their extension to additional rounds is left for future work. Nevertheless, arithmetic over the larger field gives strong reason to expect substantially higher complexities than for AES.

This also explains why the larger instances use fewer rounds. The wide-trail argument guarantees at least 25 active Sboxes over any four consecutive rounds for all instances. Since the maximum differential probability of each Sbox is $2^{-(n-2)}$, the corresponding four-round trail bound is $2^{-25(n-2)}$, compared with the target bound 2^{-16n}. This provides a margin of $9n - 50$ bits, which increases with n. On the algebraic side, the inverse Sbox has the maximal algebraic degree $n-1$, so the algebraic complexity contributed by each round also increases with n. Larger fields therefore provide greater differential and algebraic resistance per round, allowing the larger instances to attain the target security with fewer rounds. We therefore set the round counts of Lumora$(256, 16)$, Lumora$(512, 32)$, and Lumora$(1024, 64)$ to 12, 10, and 8, respectively.

4.5 Other Design Choices

The ShiftRows operation rotates row i of the 4×4 state array by i cell positions to the right. This ensures that after one round, each column contains cells originating from all four different columns of the previous state. This design paradigm follows the wide trail strategy established in AES [23,24], with the distinction that Lumora uses right rotations instead of left rotations. In a linearized representation of the state as $(s_0, \ldots, s_{15})$, ShiftRows can be implemented as $s_j \leftarrow s_{(13 \cdot j \bmod 16)}$ for $j = 0, 1, \ldots, 15$. The multiplier 13 satisfies $\gcd(13, 16) = 1$, ensuring that this mapping is a permutation of all 16 positions. Combined with MixColumns, ShiftRows guarantee full diffusion within two rounds in all variants of Lumora. This row-dependent permutation also breaks column-wise symmetry. Because MixColumns acts independently on columns, ShiftRows prevents column-local structures from remaining aligned across consecutive rounds. At the end of each round, cell-dependent constants are added to each state to destroy structural symmetry.

By combining a minimal R1CS constraint cost per Sbox with an efficient sparse MDS factorization, Lumora is uniquely optimized for zkSNARK applications. Furthermore, its strong security guarantees are grounded in decades of cryptanalytic study of AES [23,24].

5 Security Analysis

We carefully considered recent progress and new attacks in the field of algebraic cryptanalysis against AO primitives [9,10,58] to evaluate the security of the different instances of Lumora. Furthermore, the wide trail design strategy provides tight upper bounds on the maximum probability of multi-round differential trails and on the maximum correlation of multiround linear trails, thereby offering a systematic parameterization of resistance to differential and linear cryptanalysis. We now summarize Lumora's security measures for its variants against different attacks.

5.1 Algebraic Cryptanalysis

In algebraic cryptanalysis, the encryption process of a cipher is described as a large set of multivariate polynomial equations [10,18,19]. The security of a cipher against algebraic attacks is directly related to the difficulty of solving the system of polynomial equations. Gröbner bases have been used as standard representations of polynomial ideals that may allow for an efficient calculation of the variety. In [6] the authors sketched algebraic attacks relying on Gröbner bases in three basic steps: i) computing a Gröbner basis of the polynomial ideal induced from a target cipher in degree-related order, such as *degree-reverse lexicographical order*; ii) converting the Gröbner basis into *lexicographical order*, yielding a univariate polynomial equation sharing its solutions with the original system; iii) factorizing the univariate polynomial, and back-substituting its roots to derive solutions of the original system induced from the cipher. In the following we discuss the difficulty of the above steps for the Lumora family.

In the following discussion we consider an attack model using a single pair of plaintext and ciphertext. The Lumora encryption will be represented as a set of multivariate polynomials, yielding nonlinear equations when embedding a pair of plaintext and ciphertext. We discuss below two state-of-the-art methods to expressing the multivariate polynomial equations from the Lumora cipher.

5.1.1 Polynomials in 16 variables. We first take a natural approach that only considers the encryption key as variables. In this setting we can equivalently set the *initial state* as 16 variables $x_0, \ldots, x_{15}$ in $\mathbb{F}_q$, which are independent of block size and the number of rounds. As such we will consider the multivariate polynomial ring $\mathbb{F}_q[x_0, \ldots, x_{15}]$ with coefficients in the extension field $\mathbb{F}_q$ with $q = 2^n$. By incorporating $x_i^q = x_i$ for each variable, each state element $s_0, \ldots, s_{15}$ (generic state elements at any point in the cipher) can be seen as polynomials in the multivariate polynomial ring over the ideal spanned by the basis $(x_0^q - x_0, \ldots, x_{15}^q - x_{15})$. Consequently a proper definition of the system of multivariate polynomials is every state element $s_i \in \mathcal{R} = \mathbb{F}_q[x_0, \ldots, x_{15}]/\mathcal{I}$ where $\mathcal{I} = \langle x_0^q - x_0, \ldots, x_{15}^q - x_{15} \rangle$.

Below we show that it is computationally infeasible to represent a multivariate polynomial system for more than two rounds of Lumora, which clearly

indicates the intractability of solving the polynomial system. We estimate the number of monomials, denoted by d, for each polynomial in this system (in the worst case). Assume in one cell the polynomial has d monomials (excluding the constant term), denoted as $Z = Z_1 + \cdots + Z_d$, where Z_i for $i = 1, \ldots, d$ is a monomial in $\mathbb{K}$. In the SubCell, the inverse $Z^{-1} = Z^{2^n-2} = Z^{2^{n-1}+2^{n-2}+\cdots+2} = Z^{2^{n-1}} Z^{2^{n-2}} \cdots Z^2$, yielding a polynomial with d^{n-1} monomials, and the linearized polynomial expands it to a polynomial with nd^{n-1} monomials. The linear layer of the MixColumns expands the polynomial by a factor of 4 and the effect of ShiftRows operation can be ignored since the polynomials in all cells are expected to have the same number of monomials. That is to say, one round of Lumora expands a polynomial of d monomials to a new one with $4nd^{n-1}$ monomials. Starting from $d = 1$ in the first round, the number of monomials increases from 1 to $4n$; then the growing speed of d becomes exponential in each round, increasing from $4n$ to $4n(4n)^{n-1}$ in the second round, and increasing from $4n(4n)^{n-1}$ to $4n(4n(4n)^{n-1})^{n-1}$ in the third round, reaching the limit that each polynomial in $\mathcal{R}$ contains at most $2^{16(n-1)}$ monomials.

We now investigate the algebraic degree of each polynomial. The only operation affecting the degree is SubCell and the inverse of polynomials. Combined with $L(x)$ and the linear combinations of MixColumns, we expect the algebraic degree to be 2^{n-2} in average (because of the reduction by $\mathcal{I}$). After Round 1, polynomials have partial degree 2^{n-2} in only four variables but the diffusion properties of M_L ensures all 16 variables have maximum partial degree after Round 2. We will consider they have degree 2^n for simplicity. The main bottleneck in terms of performance comes from the cost of multiplying two multivariate polynomials efficiently. There exist two ways of representing polynomials, each offering a time/memory tradeoff.

Sparse representation. The polynomials are stored as objects similar to hashmaps in memory. Only the monomials whose coefficient is non-zero are represented. The multiplication of two polynomials in sparse representation follows Johnson's algorithm [44]. Given two multivariate polynomials respectively with d_1 and d_2 monomials, the time complexity is $\mathcal{O}(d_1 d_2)$ and the memory is $\mathcal{O}(d_1 d_2)$.

Dense representation. The coefficient for each monomial is stored in memory. The most efficient multiplication algorithm for two polynomials in dense representation uses FFT (Fast Fourier Transform) or NTT (Number Theoretic Transform) and Kronecker substitutions to transform into univariate forms. In theory, NTT is faster but in practice, the FFT approach performs better on common hardware for generic polynomials. The complexity is as follows. Let $P, Q \in \mathbb{K}$ with partial degrees $\deg_{x_i}(P) = \alpha_i$ and $\deg_{x_i}(Q) = \beta_i$. Using the notation $\mathcal{M}(t) = \mathcal{O}(t \log(t))$, the complexity for univariate polynomial multiplication using FFT has time and memory complexity respectively

$$\mathcal{M}\left(\prod_{i=1}^{16}(\alpha_i + \beta_i + 1)\right) \text{ and } \mathcal{O}\left(\prod_{i=1}^{16}(\alpha_i + \beta_i + 1)\right).$$

Representing the SubCell operation for $r = 2$ means to multiply two polynomials of $d_1 = (4n + 1)^{n-2}$ and $d_2 = 4n + 1$ monomials (since the linearized polynomial step $L(x)$ can be seen as repeated squarings which cost is negligible). The complexity for one such multiplication in sparse representation is $\mathcal{O}((4n + 1)^{n-1})$. For the case of $n = 16$, this reduces to $\mathcal{O}(2^{90})$ and there are 16 cells in total in $\mathbb{S}$. It also means multiplying polynomials with partial degree $2^n - 2$ in 4 variables each. In dense representation, the time complexity is estimated to be approximately $\mathcal{M}(2^{8n}) \approx \mathcal{O}(2^{8n+7})$. For $n = 16$, this comes down to $\mathcal{O}(2^{135})$

We extend the theory to practice with the help of the `FLINT` [3] and `NTL` [4] libraries in `C` and `C++` and conclude that expressing the polynomial system of two rounds of Lumora is infeasible in both representations on a regular laptop.

5.1.2 Polynomials in a larger number of state variables. To keep the degree of the multivariate polynomials low, the encryption process can be modelled by considering each round (i.e. the nonlinear and linear steps of each round) separately [19]. Below we describe how to construct a *degree-lexicographical* Gröbner basis describing the Lumora encryption process.

For each round of Lumora, we introduce a set of so-called intermediate state variables $\mathcal{X}_r := \{x_{r,0}, \ldots, x_{r,15}\}$. These variables represent the internal state of Lumora after the execution of the r-th round, the nonlinear layer by SubCell combined with the diffusion layer of MixColumns and ShiftRows. For the r-th round, these variables are the output variables; correspondingly the state variables of the $(r-1)$-th round are input variables of the r-th round. The input and the output of Lumera are modelled by plaintext variables $P := \{x_{0,0}, \ldots, x_{0,15}\}$ and ciphertext variables $C := \{x_{R_n,0}, \ldots, x_{R_n,15}\}$, respectively. For the encryption key, a set of key variables $K := \{k_0, \ldots, k_{15}\}$ is used. We denote by $\mathcal{X}$ the set of all state variables and key variables, and by $\mathcal{I} \subset \mathbb{F}_q[\mathcal{X}]$ the ideal associated with the encryption process of Lumora.

Similarly to [18], we perform the MixColumns and ShiftRows operations as multiplying a 16×16 matrix over $\mathbb{F}_q$ by a column vector. More precisely, the MixColumns operation ℓ can be equivalently represented by the matrix

$$D_{MC} = I_4 \otimes \begin{bmatrix} \alpha^{-1}+1 & \alpha^{-1} & 1 & \alpha^{-1}+1 \\ \alpha+1 & \alpha & \alpha^{-1} & \alpha^{-1} \\ \alpha & \alpha+1 & \alpha^{-1}+1 & \alpha^{-1} \\ \alpha^{-1} & \alpha^{-1} & \alpha^{-1}+1 & 1 \end{bmatrix} \in \mathbb{F}_q^{16\times 16}$$

and the ShiftRows operation π can be represented by the permutation matrix

$$D_{SR} = \left[\delta_{j,k}\right]_{0\le j,k<16}$$

where the entries $\delta_{j,k}$ are given by $\delta_{j,k} = 1$ if $k = 13 \cdot j \bmod 16$ and $\delta_{j,k} = 0$ otherwise. Then the linear transformation $\pi \circ \ell$ can thus be represented by

[3] https://flintlib.org/
[4] https://libntl.org/

multiplying the matrix $D = D_{SR} \cdot D_{MC}$ by the column vector. Recall that the operations of the round function are SubCells η, MixColumns π, ShiftRows L and adding round constants. Therefore, we can represent the r-th round of Lumora operation as follows: for $r = 1$,

$$\begin{pmatrix} \eta(x_{0,0} + k_0) \\ \eta(x_{0,1} + k_1) \\ \vdots \\ \eta(x_{0,15} + k_{15}) \end{pmatrix} + D^{-1} \cdot \begin{pmatrix} x_{1,0} + RC_{1,0} \\ x_{1,1} + RC_{1,1} \\ \vdots \\ x_{1,15} + RC_{1,15} \end{pmatrix} = 0, \tag{2}$$

for $2 \leq r < R_n$,

$$\begin{pmatrix} \eta(x_{r-1,0}) \\ \eta(x_{r-1,1} \\ \vdots \\ \eta(x_{r-1,15}) \end{pmatrix} + D^{-1} \cdot \begin{pmatrix} x_{r,0} + RC_{r,0} \\ x_{r,1} + RC_{r,1} \\ \vdots \\ x_{r,15} + RC_{r,15} \end{pmatrix} = 0, \tag{3}$$

and for $r = R_n$,

$$\begin{pmatrix} \eta(x_{r-1,0}) \\ \eta(x_{r-1,1} \\ \vdots \\ \eta(x_{r-1,15}) \end{pmatrix} + D^{-1} \cdot \begin{pmatrix} x_{r,0} + k_0 + RC_{r,0} \\ x_{r,1} + k_1 + RC_{r,1} \\ \vdots \\ x_{r,15} + k_{15} + RC_{r,15} \end{pmatrix} = 0, \tag{4}$$

where the Subcell operation $\eta(x) = L(x^{q-2}) + a_n$ with the linearized polynomial $L(x) = \sum_{i=0}^{n-1} a_i x^{2^i}$ explicitly given in Section A and $a_n = \alpha$ a primitive element of $\mathbb{F}_q$ for $q = 2^n$. Hence the r-th round of Lumora operation correspond to 16 polynomial equations, which involve 32 variables for $r = 1, 2, \ldots, R_n$. There are $16R_n$ polynomial equations with in total $16R_n$ different variables (excluding the plaintext and ciphertext variables). Similarly to the case for AES-128 discussed in [18], choosing any degree lexicographical term order, for each round operation, either a term $x_{r,j}^{q-2}$ or a term k_j^{q-2} occurs as head term of each polynomial and all the head terms are pairwise prime. As recalled in [18, Th. 1], a set G of polynomials is a Gröbner basis when all the head terms of polynomials in G are pairwise prime. Therefore, the set of polynomials f_i in $\mathbb{F}_q[\mathcal{X}]$ induced from Lumora encryption directly forms a Gröbner basis. Moreover, the ideal generated from the polynomials is zero-dimensional.

Exploiting the Gröbner Basis. We shall explore the cryptanalytic impact of the obtained zero-dimensional Gröbner basis $\mathcal{B}$ for the Lumora cipher.

First, we consider whether the obtained Gröbner basis $\mathcal{B}$ can be efficiently converted to a different, more suitable order, i.e. a lexicographical order in Step ii) recalled at the beginning of Section 5.1. Since $\mathbb{F}_q[\mathcal{X}]$ is discussed, it is more natural to consider the FGLM algorithm [31].

Denote $\mathcal{R} := \mathbb{F}_q[\mathcal{X}]$ and by $\mathcal{I}$ the ideal spanned from $\mathcal{B}$. According to [18, Cor. 1], since the Gröbner basis has head terms $x_{r,j}^{q-2}, k_j^{q-2}$, the $\mathbb{F}_q$-dimension

of the ideal $\mathcal{I}$ in $\mathcal{R}$ is given by $\dim(\mathcal{R}/\mathcal{I}) = \prod_{i=1}^{16R_n} \deg(f_i) = (2^n - 2)^{16R_n}$, where $R_n = 12, 10, 8$ for $n = 16, 32, 64$, respectively. The above vector space dimension is way too big for FGLM to perform a Gröbner basis conversion. Very recently, a new type of algebraic attack, termed the *FreeLunch approach*, was proposed in [10], which applies to several arithmetization-oriented families of permutations. The authors proposed a characterization of FreeLunch systems and conceived a dedicated algorithm for the resolution of FreeLunch systems, instead of relying on FGLM. According to [10, Th. 1], the proposed new algorithm could compute one solution x_0 induced from a FreeLunch system with time complexity $\widetilde{O}(\alpha_0(\prod_{i=1}^{m-1} \alpha_i)^\omega)$, where α_i is the degree of each polynomial and ω is the linear algebra constant. Observe that the Gröbner basis $\mathcal{B}$ associated with Lumora is a FreeLunch system with each polynomial having degree $q - 2$. As a result, the time complexity of the new algorithm proposed in [10] is in the order of $(2^n - 2)^{\omega(16R_n-1)+1}$, which exceeds the target threshold 2^{16n} when $R_n \geq 3$.

In addition, each variable x_i in $\mathcal{X}$ satisfies $x_i^q = x_i$. Adding the equations $x_i^q - x_i = 0$ to the Gröbner basis $\mathcal{B}$ will exclude all points of the variety that only exist in the closure but not in the ground field $\mathbb{F}_q$, which are more relevant to the cryptanalytic purpose. However, the resulting set of polynomials no longer forms a Gröbner basis. In this case, should the system been regular, its degree of regularity would have been given by the Macaulay bound $d_{\text{reg}} \leq 1 + \sum_{i=1}^{16R_n} (\deg(f_i) - 1)$, where f_i denotes the polynomial derived from (2)-(4) and has degree $2^n - 2$. As recalled in [18, App. A], knowledge of the degree of regularity provides a lower bound on the complexity of computing a Gröbner basis. For the set of polynomial equations induced from Lumora, there are $16R_n$ variables. As a result, there are roughly $\binom{16R_n + d_{\text{reg}}}{16R_n} = \binom{(2^n-2)16R_n}{16R_n}$ monomials, indicating a lower bound $\binom{(2^n-2)16R_n}{16R_n}^\omega$ on the complexity of deriving a Gröbner basis, which exceeds target threshold 2^{16n} when $R_n \geq 3$.

5.2 Differential and Linear Cryptanalysis

The wide trail design strategy [21] provides tight upper bounds on the maximum probability of multiround differential trails and on the maximum correlation of multiround linear trails, thereby offering a systematic parameterization of resistance to differential and linear cryptanalysis. In AES-like designs, if the permutation (ShiftRows) is chosen appropriately, a strong lower bound can be established on the number of active Sboxes in any four-round trail. Specifically, similar to the AES structure, the Lumora($16n$, n) design guarantees that the minimum number of active Sboxes in any four-round trail is at least the square of the branch number of the matrix used in the MixColumns operation. Since Lumora($16n$, n) employs a 4×4 MDS matrix, this ensures that the minimum number of active Sboxes in any four-round differential (or linear) trail is lower bounded by 25.

The cryptanalytic properties of the Sbox used in Lumora($16n$, n) stem from Nyberg's analysis of the inverse mapping [54]. When n is even, the inverse Sbox

has differential uniformity 4. In the Lumora($16n$, n) family, the underlying finite field is $\mathbb{F}_{2^n}$, with $n \in \{16, 32, 64\}$, so the differential uniformity remains 4. Moreover, it achieves the best-known nonlinearity for even n, which is $2^{n-1} - 2^{\frac{n}{2}}$. Consequently, the maximum differential probability of the Sbox is 2^{-n+2}, and the maximum absolute correlation is $2^{-\frac{n}{2}+1}$.

In general, to mount a differential cryptanalysis on an b-bit block cipher, an attacker needs to find a differential trail with probability higher than 2^{-b} [43]. A common approach for assessing a cipher's resistance to differential cryptanalysis is to provide a lower bound on the number of active Sboxes, denoted N_1, in any r-round differential characteristic. The probability of any r-round differential characteristic is upper bounded by $(\Delta_S)^{N_1}$, where Δ_S is the maximum differential probability of the Sbox in the cipher. Therefore, if a substantially reduced-round version satisfies $(\Delta_S)^{N_1} < 2^{-b}$, it is generally considered resistant to differential cryptanalysis. Similarly, the maximum absolute linear correlation of any r-round linear trail is upper bounded by $(\epsilon_S)^{N_2}$, where ϵ_S is the maximum absolute linear correlation of the Sbox and N_2 denotes the minimum number of active Sboxes in the r-round linear trail. If a reduced-round variant satisfies $(\epsilon_S)^{N_2} < 2^{-b/2}$ [43], it is considered resistant to linear cryptanalysis.

Applying this to our context, for differential cryptanalysis we require:

$$(2^{-n+2})^{N_1} < 2^{-b} = 2^{-16n} \implies N_1 > \frac{16n}{n-2}$$

and for linear cryptanalysis:

$$(2^{-n/2+1})^{N_2} < 2^{-b/2} = 2^{-8n} \implies N_2 > \frac{16n}{n-2}.$$

For the Lumora($16n$, n) family, where $n \in \{16, 32, 64\}$, it follows that any four-round variant ensures a sufficient number of active Sboxes in both differential and linear trails. Hence, the Lumora family provides a substantial security margin against both differential and linear cryptanalysis.

5.3 Square attack

In the setting over $\mathbb{F}_{2^{16}}$, one might argue that even in the ZKP framework, knowledge of 2^{16} blocks of plaintext/ciphertext pairs (for a fixed key) to the attacker is plausible. This constitutes the conditions in which the square attack [22] takes place.

Contrary to AES, Lumora is less vulnerable to the square attack (and integral attacks in general) because of the operations order, the bigger field size and the fact that the last rounds includes a MixColumns.

Let a δ-set of 2^n states that are all different in one particular cell s_0 and constant in all others. The course of the attack is as follows: by feeding the δ-set to the oracle, we are able to track their structure throughout multiple rounds. The structure of a cell w.r.t the δ-set is either *active* which means it takes all possible values from 0 to $2^n - 1$ exactly once. It can be *destructured* if we lose

the property of uniqueness of values among all 2^n cells in the δ-set but if we still verify that they all XOR to 0. Finally, it can be *unknown* if we lost all properties.

We will start at round $r = 1$ and follow the evolution of the cells in $\mathbb{S}$. We provide a visual representation of the attack in Figure 3. The first SubCell doesn't affect the active cells. They will inherently change value but uniqueness over the δ-set is preserved. The MixColumns is different: each column takes for new values a linear combination of all its elements. Since all others cells in $\mathbb{S}$ are constant, we only depend on active cells; it makes all cells in the column active. The ShiftRows operation doesn't affect the active cells but spread one in each column. In round 2, the MixColumns will spread active cells to the whole state. When entering round 3, the next MixColumns operation will turn all active cells into destructured cells, but keep in mind that all destructured cells at the same offset XOR to 0 in the δ-set. Destructured cells become unknown during the SubCell operation in round 4. The state is then XORed with the key to produce the ciphertext following the Even–Mansour setting. This is the exact moment where to conduct the attack. It requires to bruteforce all possible combinations of a key element (n bits) for an unknown cell. To find the element s_0 of the state before the SubCell in the fourth round, we need to backtrack the all the round operations by trying all combinations of an element of the key. We know a candidate for a key element is correct when after backtracking, we obtain destructured cells in the δ-set.

In the AES structure, the last round does not perform a MixColumns which makes retrieving an element from a destructured state dependent on bruteforcing exactly one element from the key i.e. 2^n operations (the distinguisher works in the destructured case). However, Lumora does have a final MixColumns which means we will need to bruteforce elements by groups of 4. The 4 stars $\star$ in Figure 3 show the dependency of four elements from the destructured state at the beginning of round 4. Thus the complexity of 4-rounds square attack for a field of size 2^n on Lumora is $\mathcal{O}(4 \times 2^{4n} \times 2^n) = \mathcal{O}(2^{5n+2})$ with the knowledge of 2^n plaintext/ciphertext pairs.

5.4 Invariant Subfield Attacks

An invariant subfield attack [5] exploits a proper subfield of the state space that is preserved by the round function, allowing the adversary to restrict the analysis to a smaller field. This issue arises only over extension fields, since prime fields have no proper subfields. For the three instantiated parameter sets Lumora$(256, 16)$, Lumora$(512, 32)$, and Lumora$(1024, 64)$, the state lies in $\mathbb{F}_{2^n}$ with $n \in \{16, 32, 64\}$, respectively, so proper subfields $\mathbb{F}_{2^d}$ exist. The inversion map preserves every subfield, and a linearized polynomial preserves $\mathbb{F}_{2^d}$ only when all its coefficients lie in $\mathbb{F}_{2^d}$. Therefore, Lumora chooses the coefficients of the affine polynomial and the round constants outside every proper subfield of $\mathbb{F}_{2^n}$. For $n = 16$, the instantiated polynomial $L(x)$ has coefficients that generate $\mathbb{F}_{2^{16}}$ as a field; in particular, values such as `0x110`, `0x481d`, and `0x81e3` do not lie in $\mathbb{F}_{2^8}$. Moreover, the round constants are also chosen outside any proper subfield.

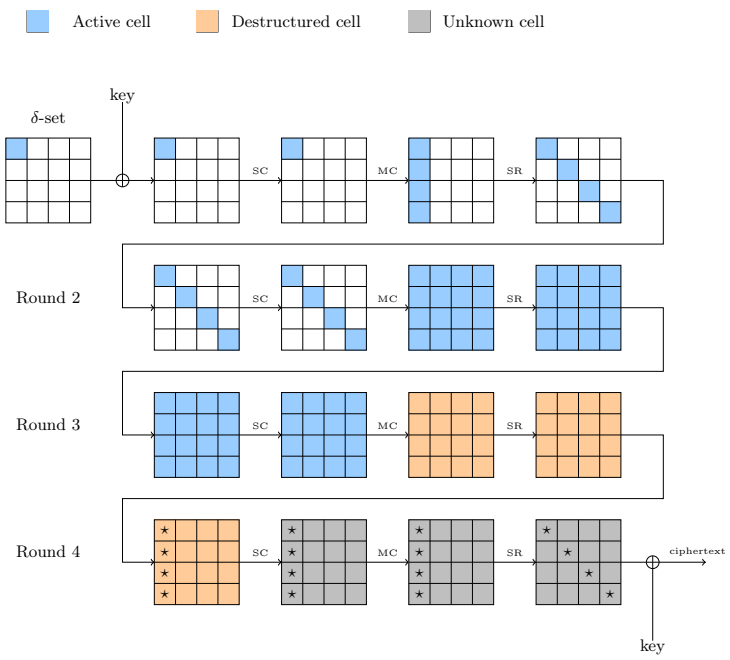


Fig. 3. Visualization of the integral attack on Lumora($16n$, n), 4 rounds.

A similar analysis applies for $n = 32$ and $n = 64$. Therefore, the Lumora family is designed to resist invariant subfield attacks.

5.5 Boomerang Attacks

In a boomerang attack [55], an attacker can join two good differential trails that each cover only part of a cipher. This can break some ciphers even if standard differential cryptanalysis does not work. One trail is for the first part of the cipher, and the other is for the second part. If the two trails have similar probability, then the attack cost is roughly the inverse of the product of the squares of their probabilities. To calculate the number of rounds sufficient to ensure that no good boomerang exists, we determine the number of rounds after which we cannot separate the cipher into two parts and find a differential for each such that the product of their probabilities is less than $2^{-b/2}$. Building on the analysis in Section 5.2, we conclude that four rounds of each Lumora instance suffice to achieve this goal.

5.6 Impossible Differential Attacks

Classical differential cryptanalysis aims to identify differentials that occur with high probability. The key candidate considered most likely is the one for which

the expected output difference appears most frequently. Impossible differential cryptanalysis takes the opposite approach, instead of looking for highly probable differences, it rules out any key that would produce an output difference known to be impossible. An impossible differential for an encryption function F is a pair $(\Delta x, \Delta y)$ such that, for every plaintext x, the relation $F(x) + F(x + \Delta x) \neq \Delta y$ holds. When such a differential exists for a reduced-round variant of a cipher, it can be leveraged to recover keys for additional rounds. The attack strategy is to choose many plaintexts with input differences consistent with the impossible differential and collect their ciphertexts. Then, by partially decrypting the extra rounds under all candidate subkeys, any subkey that produces an impossible output difference can be discarded.

In [56], it was proved that for AES there exists no truncated impossible differential spanning more than four rounds when considering the actual inverse affine Sbox. Since Lumora adopts an AES-like structure also based on the inverse Sbox over a binary extension field, it is reasonable to conclude that Lumora similarly has no truncated impossible differential extending beyond four rounds. It is also worth noting that the known impossible differential attack on seven-round AES-128 by Mala et al. [51] is considered essentially optimal for key recovery on AES-128 and has served as a foundational point for later cryptanalytic improvements such as [47]. Adapting such analyses to Lumora's wider block size and field settings would likely result in significantly higher attack complexities, consistent with our current security claims.

5.7 Higher-Order Differential Cryptanalysis

Higher-order differential cryptanalysis, developed by Knudsen [45] and Lai [46], takes advantage of properties in the polynomial representation of round-reduced symmetric primitives, especially low algebraic degree. Consider a function $F : \mathbb{F}_2^m \to \mathbb{F}_2^m$. We can define its derivative with respect to any vector $a \in \mathbb{F}_2^m$ as $D_a F(x) := F(x \oplus a) \oplus F(x)$ [46]. This concept extends to higher orders. For a k-dimensional subspace $V \subseteq \mathbb{F}_2^m$ with basis $\{a_1, \ldots, a_k\}$, the k-th order derivative is $D_V F := D_{a_1} D_{a_2} \cdots D_{a_k} F$. We also know that, for any $x \in \mathbb{F}_2^m$, $D_V F(x) = \bigoplus_{v \in V} F(x + v)$ [46] and can be easily predicted when the dimension of the considered subspace exceeds the algebraic degree of the function. More specifically, we have the following proposition

Proposition 1. *[16,46] For any function $F : \mathbb{F}_2^m \to \mathbb{F}_2^m$ and any subspace V of $\mathbb{F}_2^m$, we have*

$$\deg(D_V F) \leq max\{0, \deg(F) - \dim(V)\}.$$

Notably, for any V with $\dim(V) > \deg(F)$, we have $\bigoplus_{v \in V} F(x+v) = 0 \; \forall x \in \mathbb{F}_2^m$.

If we know that the degree of the function is strictly lower than the expected degree of a random function with similar characteristics (in particular, strictly lower than $(n-1)$ for a permutation F on $\mathbb{F}_2^n$), then the previous proposition provides us with a distinguishing attack requiring $2^{\deg(F)+1}$ data complexity. In other words, the complexity of the attacks decreases with the algebraic degree of

F, implying that it is important to understand how this quantity increases as a given round function is iterated. Lumora's resistance to higher-order differential cryptanalysis follows directly from its inverse-based Sbox over $\mathbb{F}_{2^n}$, which has algebraic degree $n-1$. Specifically, the SubCell transformation η achieves this degree for each cell in a single round. The subsequent linear layers, the 4×4 MDS MixColumns (ℓ) and the ShiftRows cell permutation (π), facilitate the rapid diffusion of these high-degree terms across the entire $16n$-bit state.

As the round function iterates, the algebraic degree of the full Lumora permutation is expected to increase rapidly. Due to Lumora's wide-block structure, however, an exact bit-level characterization of this growth is computationally prohibitive, so we do not make a precise claim about degree saturation for the full design. Nevertheless, the selected round numbers $r \in \{12, 10, 8\}$ for instances with $n \in \{16, 32, 64\}$, respectively, are anticipated to provide a substantial security margin beyond the rounds required for degree saturation. As discussed in Section 5.1, this ensures that the permutation is not only of maximum degree but also algebraically dense, thereby providing robust resistance against higher order differential cryptanalysis.

6 R1CS Cost Analysis and Comparison

This section compares Lumora with representative arithmetization-oriented permutations defined over binary extension fields, namely Vision [6], POSEIDONB, and POSEIDON2B [33]. Although Lumora is instantiated as a wide-block cipher through the Even–Mansour construction, the comparison concerns the R1CS cost of evaluating the underlying public permutations. Restricting the comparison to binary-field designs provides a common arithmetization model and avoids conflating costs arising from different underlying fields. We count the nonlinear multiplication constraints required to evaluate one full permutation over its native binary extension field $\mathbb{F}_{2^n}$. Under this cost model, additions, multiplication by public constants, Frobenius maps $x \mapsto x^{2^i}$, and fixed $\mathbb{F}_2$-linear transformations can be absorbed into the linear part of the R1CS and therefore incur no nonlinear multiplication constraints. This cost convention is particularly relevant to proof systems designed to operate efficiently over binary fields, such as Binius [25].

Note that the Lumora Sbox is defined by $S(x) = L(x^{-1}) + a_n$, where L is a fixed invertible $\mathbb{F}_2$-linear map and $a_n \in \mathbb{F}_{2^n}$ is a public constant. For an Sbox input x and a claimed output z, define the auxiliary value $w = L^{-1}(z + a_n)$. Because L^{-1} is fixed and $\mathbb{F}_2$-linear, computing w from z incurs no nonlinear multiplication constraints. The zero input is not represented by this constraint. Under the convention $y = 0$ when $x = 0$, the correct Sbox output at zero is $S(0) = L(0) + a_n = a_n$, and the value of $w = L^{-1}(a_n + a_n) = 0$. Nevertheless, $xw = 1$ reduces to $0 = 1$ when $x = 0$, so no satisfying assignment exists, even for the valid pair $(x, z) = (0, a_n)$. Consequently, the one-constraint encoding rejects any otherwise valid computation in which an Sbox input is zero. This introduces a completeness error, since an honest prover cannot prove such a computation,

but it does not weaken soundness as every satisfying assignment has $x \neq 0$, forces $w = x^{-1}$, and therefore forces the claimed output to equal $S(x)$.

If perfect completeness is required, the total inverse convention can instead be enforced by

$$x^2 w = x, \qquad x w^2 = w.$$

Since squaring is a Frobenius map, both x^2 and w^2 are obtained without nonlinear constraints. The two equations, therefore require exactly two nonlinear multiplication constraints per Sbox. For $x = 0$, they force $w = 0$; for $x \neq 0$, either equation forces $w = x^{-1}$. Thus, the system enforces $w = x^{-1}$ with the convention $w = 0$ when $x = 0$.

To quantify the completeness error of the one-constraint encoding, assume that each Sbox input is marginally uniform over $\mathbb{F}_{2^n}$. The probability that any particular input is zero is then $\Pr[x = 0] = 2^{-n}$. A Lumora permutation with r rounds evaluates $16r$ Sboxes. Therefore, by the union bound, the probability that at least one Sbox input is zero during a single permutation evaluation is at most $\frac{16r}{2^n}$. Concretely, the bound are $192/2^{16} \approx 2^{-8.4}$ for Lumora$(256, 16)$, $160/2^{32} \approx 2^{-24.7}$ for Lumora$(512, 32)$, and $128/2^{64} = 2^{-57}$ for Lumora$(1024, 64)$.

The MixColumns, ShiftRows, round constant addition, and Even–Mansour key additions in the Lumora round function consist entirely of fixed $\mathbb{F}_2$-linear operations and additions of public constants, and therefore cost exactly zero nonlinear multiplication constraints under the native binary field cost model. Hence, the Sbox layer is the sole source of nonlinear R1CS constraints. Since each round applies 16 Sboxes and each Sbox requires one constraint under the bounded-error encoding, an r-round Lumora permutation requires exactly $16r$ nonlinear multiplication constraints. Consequently, Lumora$(256, 16)$, Lumora$(512, 32)$, and Lumora$(1024, 64)$ require 192, 160, and 128 constraints, respectively. These figures apply to the sound but imperfectly complete one-constraint encoding; requiring perfect completeness doubles the costs to 384, 320, and 256 constraints, respectively.

Remark 1. For $n = 16$, the completeness error $192/2^{16} \approx 2^{-8.4}$ corresponds to a worst-case estimate of approximately one failure per 341 uniformly sampled permutation evaluations and is therefore operationally significant. The corresponding bounds for the $n = 32$ and $n = 64$ instances are substantially smaller, but they must nevertheless be included explicitly in the completeness statement of any protocol using the one-constraint encoding. In particular, the one-constraint encoding for the $n = 16$ instance should be used only when the protocol permits the prover to re-randomize the inputs. For fixed public inputs, such as a previously committed Merkle authentication path, a zero input cannot in general be eliminated by retrying without changing the statement being proved. In that setting, the two-constraint total-inverse encoding should be used to obtain perfect completeness.

Vision. Vision [6] is the closest structural comparator to Lumora, since it is the prior arithmetization-oriented design over binary extension fields whose nonlinear layer is built from the multiplicative inverse. A Vision round consists of

two half-rounds. Writing B for the fixed $\mathbb{F}_2$-linearized affine polynomial used by the design and B^{-1} for its compositional inverse, the first half-round applies $x \mapsto B^{-1}(x^{-1})$ to every state element, and the second applies $x \mapsto B(x^{-1})$ to every state element. So each half-round contributes exactly one nonlinear constraint per state element. For a state of m elements over $\mathbb{F}_{2^n}$ and r rounds, the total nonlinear cost is, therefore, $C_{\text{Vision}} = 2mr$. Appendix D of [6] gives two concrete instances: Vision Mark I ($m = 16$ over $\mathbb{F}_{2^8}$, a 128-bit state with parameters comparable to AES) and Vision Mark II ($m = 4$ over $\mathbb{F}_{2^{128}}$, a 512-bit state at 128-bit security). In sponge mode, both require $r = 10$ rounds, the minimum imposed by the round-selection rule $r = 2\max(\ell_0, \ell_1, 5)$, so $C_{\text{Vision}} = 2mr$ gives 320 constraints for Mark I and 80 for Mark II. Vision Mark-32 [8] is the field-matched comparator for Lumora$(512, 32)$ as both work over $\mathbb{F}_{2^{32}}$ and target a native binary field proof system. With state size $m = 24$ (768 state bits) and $r = 8$ rounds, it has $C_{\text{Vision Mark-32}} = 2mr = 384$ nonlinear constraints.

POSEIDONB and POSEIDON2B. POSEIDONB and POSEIDON2B [33] are binary extension field variants of Poseidon [36] and Poseidon2 [37] that follow the Hades design strategy. Both are instantiated over $\mathbb{F}_{2^n}$ for $n \in \{32, 64, 128\}$, with state cell size $m \in \{4, 6, 8, 12, 16, 24\}$. Their Sbox is the power map $x \mapsto x^d$, which permutes $\mathbb{F}_{2^n}$ if and only if $\gcd(d, 2^n - 1) = 1$. Since $d = 3$ and $d = 5$ are not invertible for the selected extension degrees, all specified instances use $d = 7$. As $x^7 = (x^4 \cdot x^2) \cdot x$, its evaluation requires two nonlinear multiplication constraints, while the squarings are free Frobenius maps. Each full (external) round applies the Sbox to all m state cells, whereas each partial (internal) round applies it to one cell. Hence, an instance with R_F external rounds and R_P internal rounds requires

$$C_{\text{Poseidon(2)b}} = 2 \cdot (m \cdot R_F + R_P)$$

nonlinear constraints. In particular, the $(n, m) = (32, 16)$ instance, which matches the field and state size of Lumora$(512, 32)$, uses $(R_F, R_P) = (8, 15)$ for POSEIDONB and $(10, 15)$ for POSEIDON2B, requiring 286 and 350 nonlinear constraints, respectively. Table 2 collects the resulting counts for the full permutation.

7 Reference Implementation

We provide a reference implementation for all instances of Lumora in native `C` available at https://www.github.com/lumora-aop/lumora. This version is optimized for `x86_64` architectures. It uses Intel(R) intrinsics[5] to perform single instructions carryless multiplications (CLMUL). This allows us to drastically reduce the number of instructions required to perform the MixColumns and SubCell operations.

Table 3 presents the performance of Lumora primitives in nanoseconds, `x86_64` instructions and CPU cycles. Although Lumora is meant to be used

[5] https://www.intel.com/content/www/us/en/docs/intrinsics-guide/index.html

Table 2. R1CS constraints for permutations over binary extension fields.The "Rounds" column gives the total number of rounds r for Lumora and Vision, and the pair (R_F, R_P) of full and partial rounds for POSEIDONB and POSEIDON2B. The reported Lumora and Vision counts do not include the additional constraints required for perfect completeness and must be doubled when perfect completeness is imposed. The reported POSEIDONB and POSEIDON2B counts already provide perfect completeness.

Primitive	**Field**	**# Cells (m)**	**State size (bits)**	**Rounds** r or (R_F, R_P)	**# R1CS constraints**
Lumora$(256, 16)$	$\mathbb{F}_{2^{16}}$	16	256	12	192
Lumora$(512, 32)$	$\mathbb{F}_{2^{32}}$	16	512	10	160
Lumora$(1024, 64)$	$\mathbb{F}_{2^{64}}$	16	1024	8	128
Vision Mark I	$\mathbb{F}_{2^8}$	16	128	10	320
Vision Mark II	$\mathbb{F}_{2^{128}}$	4	512	10	80
Vision Mark-32	$\mathbb{F}_{2^{32}}$	24	768	8	384
POSEIDONB	$\mathbb{F}_{2^{32}}$	16	512	$(8, 15)$	286
POSEIDON2B	$\mathbb{F}_{2^{32}}$	16	512	$(10, 15)$	350
POSEIDONB	$\mathbb{F}_{2^{32}}$	24	768	$(8, 15)$	414
POSEIDON2B	$\mathbb{F}_{2^{32}}$	24	768	$(10, 15)$	510
POSEIDONB	$\mathbb{F}_{2^{64}}$	8	512	$(8, 29)$	186
POSEIDON2B	$\mathbb{F}_{2^{64}}$	8	512	$(10, 29)$	218
POSEIDONB	$\mathbb{F}_{2^{64}}$	12	768	$(8, 29)$	250
POSEIDON2B	$\mathbb{F}_{2^{64}}$	12	768	$(10, 29)$	298
POSEIDONB	$\mathbb{F}_{2^{128}}$	4	512	$(8, 58)$	180
POSEIDON2B	$\mathbb{F}_{2^{128}}$	4	512	$(8, 58)$	180
POSEIDONB	$\mathbb{F}_{2^{128}}$	6	768	$(8, 58)$	212
POSEIDON2B	$\mathbb{F}_{2^{128}}$	6	768	$(8, 58)$	212

as a one-way function, we additionally implemented and benchmarked the corresponding inverse primitives of SubCell (η), MixColumns (ℓ) and ShiftRows (π); these are denoted by the superscript -1 in the relevant column headers (e.g., η^{-1}, ℓ^{-1}, and π^{-1}). We also evaluate the performance on one round as well as on the full encryption and decryption process, each on one $16n$-bit block.

As seen in Table 3, the SubCell operation represents the overwhelming majority of the total computations in a round. For many ciphers, the Sbox is usually precomputed and represented as a Lookup Table (LUT) in memory. Substituting an element comes down to the cost of accessing said table. The size of the extension fields being used in Lumora prevents this method and instead, the element substituting for in the Sbox must be computed on the go. With the exception of Lumora16 for which we provide both a LUT and an arithmetic Sbox primitive. For the arithmetic version of SubCell, it can essentially be reduced (cost-wise) to computing the multiplicative inverse over $\mathbb{F}_{2^n}$. Many fast algorithms exist such as the Extended Euclidean Algorithm (EEA) or the Itoh-Tsujii algorithm and while the latter is often preferred for its better performance overall, the

Table 3. Lumora performance in nanoseconds (ns), `x86_64` instructions (inst.), and CPU cycles (Intel 13th Gen i5-1345U at 4.7GHz).

n	Metric	η	ℓ	π	η^{-1}	ℓ^{-1}	π^{-1}	Round	Round^{-1}	Enc.	Dec.
16	ns	1,276	4	≤ 2	1,312	5	≤ 2	1,279	1,315	8,817	8,477
	inst.	8,859	39	14	8,841	42	14	8,910	8,900	107,241	107,729
	cycles	3,182	10	4	3,294	12	4	3,196	3,299	21,975	21,125
16 (w. LUT)	ns	9	4	≤ 2	9	5	≤ 2	18	18	161	252
	inst.	81	39	14	81	42	14	129	132	1,904	1,900
	cycles	23	10	4	23	12	4	44	47	400	623
32	ns	2,261	5	≤ 2	2,383	7	≤ 2	2,272	2,397	13,392	13,319
	inst.	16,826	41	12	17,043	42	12	16,878	17,081	170,728	173,258
	cycles	5,625	12	5	5,928	18	5	5,646	5,972	33,334	33,133
64	ns	4,330	5	≤ 2	4,320	6	≤ 2	4,395	4,324	19,816	19,711
	inst.	33,409	49	18	33,387	60	18	33,476	33,765	268,221	270,668
	cycles	10,873	13	5	10,687	15	5	10,947	10,772	49,368	49,047

EEA method offered better times and lower number of instructions in practice. We explain this through the use of the CLMUL instructions since EEA requires multiplications not over $\mathbb{F}_{2^n}$ but over $\mathbb{F}_2[X]$. This means we avoid the modular reduction step and save a lot of time.

We emphasize that the values obtained in Table 3 represent an average since the exact metrics might vary, most likely due to ambient noise in the computer from other processes. It is especially the case for the SubCell (using a LUT), MixColumns, and ShiftRows primitives that consist in few CPU instructions: it makes them hard to measure precisely when it comes down to a few nanoseconds, and we may expect bigger variations on different hardware.

8 Comparison in the FAEST-EM-256 Framework

In zkSNARK applications, a symmetric primitive is represented as an arithmetic circuit, and the number of multiplications, equivalently, the number of nonlinear constraints in the R1CS model, is a central efficiency metric. In this section, we focus on the setting relevant to one of our target applications, namely FAEST-EM-256 [13]. Our goal is to evaluate the effect of replacing the underlying one-way function (OWF) within the same Even–Mansour based signature framework.

FAEST [13] is a post-quantum digital signature scheme selected for the third round of the NIST call for Additional Digital Signature Schemes[6]. Its security relies on standard cryptographic constructions such as AES and SHA3. A FAEST signature consists of a Non-Interactive Zero-Knowledge (NIZK) proof that uses AES as an OWF. Furthermore, FAEST offers variants of their signature scheme, three of which base their OWF on an Even–Mansour construction for block sizes

[6] https://csrc.nist.gov/Projects/pqc-dig-sig/round-3-additional-signatures

of respectively 128, 192 and 256 bits. These design similarities make FAEST-EM-256 a natural candidate for comparison.

FAEST-EM-256's OWF has block length 256 bits but since AES has a fixed 128-bit block length, the designers divide the block into two halves, each of which is then processed by AES. We compare against FAEST's AVX optimized implementation for `x86_64` architecture in `C++`[7] as their use of Intel(R) intrinsics for the OWF makes it deeply relatable to ours.

Table 4. Comparison of Lumora(256, 16) and FAEST-EM-256's OWF execution times (in nanoseconds on a 13th Gen Intel(R) Core(TM) i5-1345U), witness lengths, and projected signature sizes.

OWF \ Metric	Single Round (ns)	Encryption (ns)	Witness Length	Sig. Size (Small)	Sig. Size (Fast)
FAEST-EM-256	7	152	2688 bits	17,984 B	23,476 B
Lumora(256, 16)	18	161	2304 bits	16,928 B	21,940 B

The values in Table 4 reports the different runtimes for Lumora's and FAEST-EM-256's OWF. We did not consider the key generation procedure for the round evaluation of FAEST. We explain its speed by the use of AES dedicated instructions. The encryption metric has been measured on one 256-bit block for each OWF. We explain similar performance for the one block encryption measurement in the round key generation required by AES that is the OWF's bottleneck while Lumora is designed with the use of round constants.

We now estimate the signature size of the Lumora(256, 16) instantiation within the FAEST-EM-256 framework. Since Lumora(256, 16) has 16 Sboxes in each of its 12 rounds, the encryption layer contains $S_{\text{enc}} = 192$ inverse-based Sbox evaluations. To reduce the QuickSilver witness size, we use the finite field norm idea from FAEST [13] used for AES and Rijndael. Rather than storing the full inverse output of every Sbox, we can alternate between full and compressed representations across rounds. In odd rounds, we record the full 16-bit Sbox output. In even rounds, we record only the subfield norm $c = N_{\mathbb{F}_{2^{16}}/\mathbb{F}_{2^8}}(a^{-1}) \in \mathbb{F}_{2^8}$, so that only 8 bits are recorded there and a^{-1} can be reconstructed as $a^{2^8} \cdot c$. This yields an average witness cost of 12 bits per Sbox, and hence the total witness length of 2304 bits.

For this preliminary instantiation, we keep the VOLE and commitment parameters unchanged. In particular, we retain the security parameter $\lambda = 256$, the padding parameter $B = 64$, the number of leaf-commitment blocks $n_{\text{leafcom}} = 2$, and the GGM-tree opening threshold $T_{\text{open}} \in \{218, 234\}$ for the small and fast variants, respectively. We also keep the number of small-VOLE instances $\tau \in \{22, 32\}$ from FAEST-EM-256s/f, and replace only the OWF-specific witness length by the Lumora-based value $\ell = 2304$.This reduces the witness length

[7] https://github.com/faest-sign/faest-avx

by $2688 - 2304 = 384$ bits$= 48$ bytes. Using the FAEST signature size formula

$$|\sigma| = \tau \cdot (\ell + 3\lambda + B) + T_{\text{open}} \cdot \lambda + n_{\text{leafcom}} \cdot \lambda \cdot \tau + \lambda + 128 + 32,$$

replacing the original FAEST-EM-256 witness length of 2688 bits with the Lumora-based value of 2304 bits reduces the signature size by 48τ bytes. Consequently, the projected signature sizes decrease from 17,984 to 16,928 bytes for the small parameter set, and from 23,476 to 21,940 bytes for the fast parameter set.

9 Conclusion

In R1CS-based zkSNARKs, the dominant cost metric is the number of nonlinear constraints, which corresponds directly to the number of multiplication gates needed to arithmetize the primitive's circuit. Lumora is designed first and foremost as a wide, algebraically secure permutation for zkSNARK applications. To eliminate the constraint overhead typically associated with complex nonlinear key schedules, this permutation is subsequently instantiated as a block cipher via the single-key Even-Mansour paradigm, rendering it highly optimized for MPC-in-the-Head signature schemes like FAEST [13].

Crucially, each instance of Lumora follows a unified AES-like SPN structure, with the block size as the main varying parameter. More specifically, Lumora$(16n, n)$ is defined over the binary extension field $\mathbb{F}_{2^n}$. In this paper, we primarily focus on $n \in \{16, 32, 64\}$ because these values facilitate efficient software implementations utilizing tower fields with extension degree 2. Nevertheless, additional Lumora variants are possible by choosing an appropriate primitive polynomial for other even extension degrees n within this universal structure. Table 5 summarizes several such candidate parameter sets and their defining polynomials.

We have also briefly positioned Lumora among other zkSNARK-friendly permutations, with a particular emphasis on designs operating over binary extension fields. Finally, we evaluated the integration of Lumora into the FAEST-EM-256 framework. We observed that reducing the witness length from the original 2688 bits to the Lumora-based 2304 bits decreases the overall signature size by 48τ bytes. This successfully drops the projected signature sizes for the small and fast parameter sets to 16,928 bytes and 21,940 bytes, down from 17,984 and 23,476 bytes, respectively.

A Parameters for Lumora Instances

Lumora(256, 16)*:* This variant operates over $\mathbb{F}_{2^{16}}$ defined by the primitive polynomial

$$t(x) = x^{16} + x^{12} + x^3 + x + 1.$$

Using the standard basis $\{1, \alpha, \alpha^2, \ldots, \alpha^{15}\}$, the block matrix M_L (see (1)) yields the linearized permutation polynomial $L(x) = \sum_{i=0}^{15} a_i x^{2^i}$ with coefficients a_i for $i = 0, 1, \ldots, 15$ (in hexadecimal) as follows:

0110	481d	81e3	5b63	0a75	b3b4	7305	6ab7
b846	665c	9e0c	8df6	d2b8	4754	4c6b	2689

Lumora(512, 32)*:* This variant operates over $\mathbb{F}_{2^{32}}$ defined by the primitive polynomial

$$t(x) = x^{32} + x^{22} + x^2 + x + 1.$$

Using the standard basis $\{1, \alpha, \alpha^2, \ldots, \alpha^{31}\}$, the block matrix M_L yields the linearized polynomial that induces a permutation over $\mathbb{F}_{2^{32}}$ with coefficients (in hexadecimal) as follows:

00010100	98407f3d	ebe5725d	b555dd12	4b6a6887	f9f1bb53	a0911229	50c872a5
d52f5c90	ab6fa830	e40f6b47	2a1037f0	06fabc07	8108bace	565e375d	ed01471e
128d1dc0	bc81b771	b5342f97	ba41cb57	ccb1b298	a3a5ede8	1ba6e3a9	7c8d65b0
83339c42	a32274c2	2182efac	e25f8bc1	014d251e	6960a6fb	3a899bfc	10eebe5b

Lumora(1024, 64)*:* This variant operates over $\mathbb{F}_{2^{64}}$ defined by the primitive polynomial

$$t(x) = x^{64} + x^4 + x^3 + x + 1.$$

Using the standard basis $\{1, \alpha, \alpha^2, \ldots, \alpha^{63}\}$, the block matrix M_L yields the linearized polynomial that induces a permutation over $\mathbb{F}_{2^{64}}$ with coefficients (in hexadecimal) as follows:

0000000100010000	cb4b4ccccb4d7ff4	1704aa0e9d5470f1	f64d358fa5affe10
16799aa18ef8aab3	1d39583ee9c9c595	77bf09c61b0198ef	8fca98dec759b60a
d908688f15b807c9	f387d328612e1c03	a6795682d21d5016	5c1ff71b4d40b975
edc0200582fcebae	d7d69b0fcc2acdc0	930da19bc0035c41	2f791bc1bcd3d7da
596c753d9e94594e	d1d52602ee1cfb2d	457058fef40aab29	58c75d2f9abe4a9c
ec7993977a5af8f6	68748fe6f1f0b030	d30d2a610e6275db	dd47c8e1a5273609
ffb2c7a2d84cded9	9c9bbf4718f9e33d	d993052cbe285705	4198a651712267ec
71e765ce6c9b137e	d9faa819ac1e8a79	82b32df04ef8d9be	97973778c6fbb468
6fc0aa7515d58f75	b90728636169c811	b32c52a6696bb9a2	01b77729967f3e27
eafa7889e055b956	7a7a62f0a05ab4e9	89f25f2b3934fd60	94ff1351b317b87c
0e58813cc4790e43	bab6af9b32170b58	0e8c5614d0d89b93	b50659b392f5411d
6195b53b420d9fa0	70828cb616d65e03	3e509b8d91f161e9	321175625558a8f0
b84f8619b5571ae6	07108210a4a7dbcb	ebdeaa89faa7b5d0	ac91438fe3e9f9a2
89b222e69e1290b7	c45fb2e6140b6d53	d2d5605a7a850ee9	4ba23d7e72953252
f470b2482e349ccc	a9b1630a6a7f784b	e9678144af9266db	2f9d1cd3ab360ae4
00909bad5f8425c4	b6a6ae4820a9726a	536d781cb811b874	7fe7198380292a9b

Table 5. Candidate parameter sets and defining primitive polynomials for additional Lumora variants.

Extension Degree (n)	Block Size	Defining Primitive Polynomial
8	128 bits	$x^8 + x^4 + x^3 + x^2 + 1$
12	192 bits	$x^{12} + x^8 + x^2 + x + 1$
20	320 bits	$x^{20} + x^6 + x^4 + x + 1$
24	384 bits	$x^{24} + x^7 + x^2 + x + 1$
36	576 bits	$x^{36} + x^{14} + x^4 + x + 1$
40	640 bits	$x^{40} + x^{35} + x^2 + x + 1$
48	768 bits	$x^{48} + x^{28} + x^3 + x + 1$
56	896 bits	$x^{56} + x^{42} + x^2 + x + 1$

B Word Operation of Lumora($16n$, n)

Recall that the Sbox S in Lumora is defined by the composition of the inverse function and an affine transformation. More precisely, we have

$$S(y) = A \circ \sigma(y), \text{ where } \sigma(y) = y^{-1} \text{ and } A(y) = L(y) + a_n.$$

In each round, the nonlinear SubCell layer η is followed by the linear MixColumns layer ℓ and the ShiftRows operation π. Thus, the round function of Lumora($16n$, n) is given by $\pi \circ \ell \circ \eta$, which we will now express more compactly at the word level in order to facilitate analysis and implementation.

To describe this round function at the word level, we consider its action on a single column of the state. For $\hat{y} = (y_i, y_{i+1}, y_{i+2}, y_{i+3}) \in (\mathbb{F}_{2^n})^4$, the word operation W is defined as

$$W(\hat{y}) = \begin{bmatrix} \alpha^{-1}+1 & \alpha^{-1} & 1 & \alpha^{-1}+1 \\ \alpha+1 & \alpha & \alpha^{-1} & \alpha^{-1} \\ \alpha & \alpha+1 & \alpha^{-1}+1 & \alpha^{-1} \\ \alpha^{-1} & \alpha^{-1} & \alpha^{-1}+1 & 1 \end{bmatrix} \begin{bmatrix} A(y_i^{-1}) \\ A(y_{i+1}^{-1}) \\ A(y_{i+2}^{-1}) \\ A(y_{i+3}^{-1}) \end{bmatrix}.$$

Equivalently, using the factorization $M = M_1 M_2 M_3 M_4$ of the MDS matrix, we can write

$$W(\hat{y}) = M_1 \cdot (M_2 \cdot (M_3 \cdot (M_4 \cdot \hat{b}))), \quad \text{where} \quad \hat{b} = \begin{bmatrix} A(y_i^{-1}) \\ A(y_{i+1}^{-1}) \\ A(y_{i+2}^{-1}) \\ A(y_{i+3}^{-1}) \end{bmatrix}.$$

This word level description offers an alternative representation of the round function that can be useful for both analysis and implementation.

References

1. Alagic, G., Bai, C., Katz, J., Majenz, C.: Post-Quantum Security of the Even-Mansour Cipher. In: Advances in Cryptology – EUROCRYPT 2022. pp. 458–487 (2022). https://doi.org/10.1007/978-3-031-07082-2_17

2. Albrecht, M., Grassi, L., Rechberger, C., Roy, A., Tiessen, T.: MiMC: Efficient Encryption and Cryptographic Hashing with Minimal Multiplicative Complexity. In: Advances in Cryptology – ASIACRYPT 2016. pp. 191–219 (2016). https://doi.org/10.1007/978-3-662-53887-6_7
3. Albrecht, M.R., Cid, C., Grassi, L., Khovratovich, D., Lüftenegger, R., Rechberger, C., Schofnegger, M.: Algebraic Cryptanalysis of STARK-Friendly Designs: Application to MARVELlous and MiMC. In: Advances in Cryptology – ASIACRYPT 2019. pp. 371–397 (2019). https://doi.org/10.1007/978-3-030-34618-8_13
4. Albrecht, M.R., Grassi, L., Perrin, L., Ramacher, S., Rechberger, C., Rotaru, D., Roy, A., Schofnegger, M.: Feistel Structures for MPC, and More. In: Computer Security – ESORICS 2019. pp. 151–171 (2019). https://doi.org/10.1007/978-3-030-29962-0_8
5. Aly, A., Ashur, T., Ben-Sasson, E., Dhooghe, S., Szepieniec, A.: Design of Symmetric-Key Primitives for Advanced Cryptographic Protocols. IACR Transactions on Symmetric Cryptology **2020**(3), 1–45 (Sep 2020). https://doi.org/10.13154/tosc.v2020.i3.1-45
6. Aly, A., Ashur, T., Ben-Sasson, E., Dhooghe, S., Szepieniec, A.: Design of Symmetric-Key Primitives for Advanced Cryptographic Protocols. IACR Transactions on Symmetric Cryptology pp. 1–45 (Sep 2020). https://doi.org/10.46586/tosc.v2020.i3.1-45, https://tosc.iacr.org/index.php/ToSC/article/view/8695
7. Ashur, T., Dhooghe, S.: MARVELlous: a STARK-friendly family of cryptographic primitives. Cryptology ePrint Archive, Paper 2018/1098 (2018), https://eprint.iacr.org/2018/1098
8. Ashur, T., Mahzoun, M., Posen, J., Šijačić, D.: Vision mark-32: ZK-friendly hash function over binary tower fields. Cryptology ePrint Archive, Paper 2024/633 (2024), https://eprint.iacr.org/2024/633
9. Bariant, A., Boeuf, A., Briaud, P., Hostettler, M., Øygarden, M., Raddum, H.: Improved Resultant Attack Against Arithmetization-Oriented Primitives. In: Advances in Cryptology – CRYPTO 2025. pp. 335–367 (2025). https://doi.org/10.1007/978-3-032-01901-1_11
10. Bariant, A., Boeuf, A., Lemoine, A., Manterola Ayala, I., Øygarden, M., Perrin, L., Raddum, H.: The algebraic freelunch: Efficient gröbner basis attacks against arithmetization-oriented primitives. In: Advances in Cryptology – CRYPTO 2024. p. 139–173 (2024). https://doi.org/10.1007/978-3-031-68385-5_5
11. Baum, C., Beullens, W., Braun, L., Delpech de Saint Guilhem, C., Klooß, M., Majenz, C., Mukherjee, S., Orsini, E., Ramacher, S., Rechberger, C., Roy, L., Scholl, P.: Shorter, Tighter, FAESTer: Optimizations and Improved (QROM) Analysis for VOLE-in-the-Head Signatures. In: Advances in Cryptology – CRYPTO 2025. pp. 124–156 (2025). https://doi.org/10.1007/978-3-032-01887-8_5
12. Baum, C., Beullens, W., Mukherjee, S., Orsini, E., Ramacher, S., Rechberger, C., Roy, L., Scholl, P.: One Tree to Rule Them All: Optimizing GGM Trees and OWFs for Post-Quantum Signatures. In: Advances in Cryptology – ASIACRYPT 2024. pp. 463–493 (2025). https://doi.org/10.1007/978-981-96-0875-1_15
13. Baum, C., Braun, L., de Saint Guilhem, C.D., Klooß, M., Majenz, C., Mukherjee, S., Ramacher, S., Rechberger, C., Orsini, E., Roy, L., et al.: FAEST: Algorithm Specifications (Version 2.0). Tech. rep., FAEST Project (2024), https://faest.info/faest-spec-v2.0.pdf, accessed: 2026-03-05
14. 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. pp. 103–128 (2019). https://doi.org/10.1007/978-3-030-17653-2_4

15. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: The KECCAK reference (Jan 2011), http://keccak.noekeon.org/
16. Boura, C., Naya-Plasencia, M. (eds.): Symmetric Cryptography, Volume 2: Cryptanalysis and Future Directions. John Wiley & Sons (2023)
17. Bouvier, C., Briaud, P., Chaidos, P., Perrin, L., Salen, R., Velichkov, V., Willems, D.: New Design Techniques for Efficient Arithmetization-Oriented Hash Functions: `Anemoi` Permutations and `Jive` Compression Mode. In: Advances in Cryptology – CRYPTO 2023. pp. 507–539 (2023). https://doi.org/10.1007/978-3-031-38548-3_17
18. Buchmann, J., Pyshkin, A., Weinmann, R.P.: A Zero-Dimensional Gröbner Basis for AES-128. In: et al., H. (ed.) Fast Software Encryption, vol. 4047, pp. 78–88. Springer Berlin Heidelberg, Berlin, Heidelberg (2006). https://doi.org/10.1007/11799313_6, http://link.springer.com/10.1007/11799313_6
19. Cid, C., Weinmann, R.P.: Block Ciphers: Algebraic Cryptanalysis and Gröbner Bases, pp. 307–327. Springer Berlin Heidelberg, Berlin, Heidelberg (2009). https://doi.org/10.1007/978-3-540-93806-4_17
20. Daemen, J.: Limitations of the Even-Mansour construction. In: Advances in Cryptology –ASIACRYPT '91. pp. 495–498 (1993). https://doi.org/10.1007/3-540-57332-1_46
21. Daemen, J.: Cipher and hash function design, strategies based on linear and differential cryptanalysis, PhD Thesis. K.U.Leuven (1995), http://jda.noekeon.org/
22. Daemen, J., Knudsen, L., Rijmen, V.: The block cipher square. Lecture Notes in Computer Science **1267** (10 1998). https://doi.org/10.1007/BFb0052343
23. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Information Security and Cryptography, Springer (2002). https://doi.org/10.1007/978-3-662-04722-4
24. Daemen, J., Rijmen, V.: The Design of Rijndael: The Advanced Encryption Standard (AES). Information Security and Cryptography, Springer Berlin Heidelberg (2020). https://doi.org/10.1007/978-3-662-60769-5
25. Diamond, B.E., Posen, J.: Succinct arguments over towers of binary fields. In: Fehr, S., Fouque, P. (eds.) Advances in Cryptology - EUROCRYPT 2025 - 44th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Madrid, Spain, May 4-8, 2025, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 15604, pp. 93–122. Springer (2025). https://doi.org/10.1007/978-3-031-91134-7_4, https://doi.org/10.1007/978-3-031-91134-7_4
26. Dobraunig, C., Kales, D., Rechberger, C., Schofnegger, M., Zaverucha, G.: Shorter Signatures Based on Tailor-Made Minimalist Symmetric-Key Crypto. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. pp. 843–857. CCS '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3548606.3559353
27. Dunkelman, O., Keller, N., Shamir, A.: Minimalism in Cryptography: The Even-Mansour Scheme Revisited. In: Advances in Cryptology – EUROCRYPT 2012. pp. 336–354 (2012). https://doi.org/10.1007/978-3-642-29011-4_21
28. Eichlseder, M., Grassi, L., Lüftenegger, R., Øygarden, M., Rechberger, C., Schofnegger, M., Wang, Q.: An Algebraic Attack on Ciphers with Low-Degree Round Functions: Application to Full MiMC. In: Advances in Cryptology – ASIACRYPT 2020. pp. 477–506 (2020). https://doi.org/10.1007/978-3-030-64837-4_16
29. Ekdahl, P., Johansson, T., Maximov, A., Yang, J.: A new SNOW stream cipher called SNOW-V. IACR Transactions on Symmetric Cryptology **2019, Issue 3**, 1–42 (2019). https://doi.org/10.13154/tosc.v2019.i3.1-42

30. Even, S., Mansour, Y.: A construction of a cipher from a single pseudorandom permutation. In: Advances in Cryptology – ASIACRYPT '91. pp. 210–224 (1993). https://doi.org/10.1007/3-540-57332-1_17
31. Faugère, J.C., Gianni, P., Lazard, D., Mora, T.: Efficient computation of zero-dimensional Gröbner bases by change of ordering. J. Symb. Comput. **16**(4), 329–344 (oct 1993). https://doi.org/10.1006/jsco.1993.1051
32. Grassi, L., Hao, Y., Rechberger, C., Schofnegger, M., Walch, R., Wang, Q.: Horst Meets Fluid-SPN: Griffin for Zero-Knowledge Applications. In: Advances in Cryptology – CRYPTO 2023. pp. 573–606 (2023). https://doi.org/10.1007/978-3-031-38548-3_19
33. Grassi, L., Khovratovich, D., Koschatko, K., Rechberger, C., Schofnegger, M., Schröppel, V., Wu, Z.: Poseidon(2)b: Binary Field Versions of Poseidon/Poseidon2. IACR Communications in Cryptology **2**(4) (2026). https://doi.org/10.62056/a66ce0zn4
34. Grassi, L., Khovratovich, D., Lüftenegger, R., Rechberger, C., Schofnegger, M., Walch, R.: Reinforced Concrete: A Fast Hash Function for Verifiable Computation. In: Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. p. 1323–1335. CCS '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3548606.3560686
35. Grassi, L., Khovratovich, D., Lüftenegger, R., Rechberger, C., Schofnegger, M., Walch, R.: Monolith: Circuit-Friendly Hash Functions with New Nonlinear Layers for Fast and Constant-Time Implementations. IACR Transactions on Symmetric Cryptology **2024**(3), 44–83 (09 2024). https://doi.org/10.46586/tosc.v2024.i3.44-83
36. Grassi, L., Khovratovich, D., Rechberger, C., Roy, A., Schofnegger, M.: Poseidon: A New Hash Function for Zero-Knowledge Proof Systems. In: 30th USENIX Security Symposium (USENIX Security 21). pp. 519–535. USENIX Association (Aug 2021), https://www.usenix.org/system/files/sec21-grassi.pdf
37. Grassi, L., Khovratovich, D., Schofnegger, M.: Poseidon2: A Faster Version of the Poseidon Hash Function. In: Progress in Cryptology - AFRICACRYPT 2023. pp. 177–203 (2023). https://doi.org/10.1007/978-3-031-37679-5_8
38. Grassi, L., Lüftenegger, R., Rechberger, C., Rotaru, D., Schofnegger, M.: On a Generalization of Substitution-Permutation Networks: The HADES Design Strategy. In: Advances in Cryptology – EUROCRYPT 2020. pp. 674–704 (2020). https://doi.org/10.1007/978-3-030-45724-2_23
39. Groth, J.: On the Size of Pairing-Based Non-interactive Arguments. In: Advances in Cryptology – EUROCRYPT 2016. pp. 305–326 (2016). https://doi.org/10.1007/978-3-662-49896-5_11
40. Gupta, K.C., Pandey, S.K., Samanta, S.: Construction of Recursive MDS Matrices Using DLS Matrices. In: Progress in Cryptology - AFRICACRYPT 2022. pp. 3–27 (2022). https://doi.org/10.1007/978-3-031-17433-9_1
41. Gupta, K.C., Pandey, S.K., Samanta, S.: FUTURE: A Lightweight Block Cipher Using an Optimal Diffusion Matrix. In: Progress in Cryptology - AFRICACRYPT 2022. pp. 28–52 (2022). https://doi.org/10.1007/978-3-031-17433-9_2
42. Ha, J., Hwang, S., Lee, J., Park, S., Son, M.: `Polocolo`: A ZK-Friendly Hash Function Based on S-Boxes Using Power Residues. In: Advances in Cryptology – EUROCRYPT 2025. pp. 303–332 (2025). https://doi.org/10.1007/978-3-031-91134-7_11
43. Heys, H.M.: A tutorial on linear and differential cryptanalysis. Cryptologia **26**(3), 189–221 (2002). https://doi.org/10.1080/0161-110291890885
44. Johnson, S.C.: Sparse polynomial arithmetic. ACM SIGSAM Bulletin **8**(3), 63–71 (1974). https://doi.org/10.1145/1086837.1086847

45. Knudsen, L.R.: Truncated and higher order differentials. In: Fast Software Encryption. pp. 196–211 (1995). https://doi.org/10.1007/3-540-60590-8_16
46. Lai, X.: Higher Order Derivatives and Differential Cryptanalysis, pp. 227–233. Springer US, Boston, MA (1994). https://doi.org/10.1007/978-1-4615-2694-0_23
47. Leurent, G., Pernot, C.: New Representations of the AES Key Schedule. In: Advances in Cryptology – EUROCRYPT 2021. pp. 54–84 (2021). https://doi.org/10.1007/978-3-030-77870-5_3
48. Lidl, R., Niederreiter, H.: Finite Fields. Cambridge University Press, Cambridge, 2nd edn. (1997)
49. Liu, F., Mahzoun, M., Øygarden, M., Meier, W.: Algebraic Attacks on RAIN and AIM Using Equivalent Representations. IACR Transactions on Symmetric Cryptology **2023**(4), 166–186 (Dec 2023). https://doi.org/10.46586/tosc.v2023.i4.166-186
50. MacWilliams, F., Sloane, N.: The Theory of Error Correcting Codes. North-Holland Publishing Co., Amsterdam-New York-Oxford (1977)
51. Mala, H., Dakhilalian, M., Rijmen, V., Modarres-Hashemi, M.: Improved Impossible Differential Cryptanalysis of 7-Round AES-128. In: Progress in Cryptology - INDOCRYPT 2010. pp. 282–291 (2010). https://doi.org/10.1007/978-3-642-17401-8_20
52. Manterola Ayala, I., Phanse, A., Bouvier, C.: STAP Zoo. https://stap-zoo.com/ (2023), Maintained by the STAP-Zoo Team and Zoo-keepers
53. Merz, S.P., Àlex Rodríguez García: Skipping Class: Algebraic Attacks exploiting weak matrices and operation modes of Poseidon2(b). Cryptology ePrint Archive, Paper 2026/306 (2026), https://eprint.iacr.org/2026/306
54. Nyberg, K.: Differentially uniform mappings for cryptography. In: Advances in Cryptology – EUROCRYPT '93. pp. 55–64 (1994). https://doi.org/10.1007/3-540-48285-7_6
55. Wagner, D.: The Boomerang Attack. In: Fast Software Encryption. pp. 156–170 (1999). https://doi.org/10.1007/3-540-48519-8_12
56. Wang, Q., Jin, C.: Upper bound of the length of truncated impossible differentials for AES. Designs, Codes and Cryptography **86**(7), 1541–1552 (Jul 2018). https://doi.org/10.1007/s10623-017-0411-z
57. Yang, H.S., Zheng, Q.X., Yang, J.: Algebraic Cryptanalysis of AO Primitives Based on Polynomial Decomposition Applications to Rain and Full AIM-I/III/V. Cryptology ePrint Archive, Paper 2025/981 (2025), https://eprint.iacr.org/2025/981
58. Yang, H.S., Zheng, Q.X., Yang, J., Liu, Q.F., Tang, D.: A new security evaluation method based on resultant for arithmetic-oriented algorithms. In: Advances in Cryptology – ASIACRYPT 2024. pp. 457–489 (2025). https://doi.org/10.1007/978-981-96-0941-3_15