AIM: Symmetric Primitive for Shorter Signatures with Stronger Security*
(Full Version)

Seongkwang Kim[1**], Jincheol Ha[2**], Mincheol Son[2], Byeonghak Lee[1†], Dukjae Moon[1], Joohee Lee[3], Sangyub Lee[1], Jihoon Kwon[1], Jihoon Cho[1], Hyojin Yoon[1], and Jooyoung Lee[2‡]

[1] Samsung SDS, Seoul, Korea,
{sk39.kim,byghak.lee,dukjae.moon,sangyub0.lee,jihoon.kwon,jihoon1.cho,hj1230.yoon}@samsung.com
[2] KAIST, Daejeon, Korea,
{smilecjf,encrypted.def,hicalf}@kaist.ac.kr
[3] Sungshin Women's University, Seoul, Korea
jooheelee@sungshin.ac.kr

Abstract. Post-quantum signature schemes based on the MPC-in-the-Head (MPCitH) paradigm are recently attracting significant attention as their security solely depends on the one-wayness of the underlying primitive, providing diversity for the hardness assumption in post-quantum cryptography. Recent MPCitH-friendly ciphers have been designed using simple algebraic S-boxes operating on a large field in order to improve the performance of the resulting signature schemes. Due to their simple algebraic structures, their security against algebraic attacks should be comprehensively studied.

In this paper, we refine algebraic cryptanalysis of power mapping based S-boxes over binary extension fields, and cryptographic primitives based on such S-boxes. In particular, for the Gröbner basis attack over $\mathbb{F}_2$, we experimentally show that the exact number of Boolean quadratic equations obtained from the underlying S-boxes is critical to correctly estimate the theoretic complexity based on the degree of regularity. Similarly, it turns out that the XL attack might be faster when all possible quadratic equations are found and used from the S-boxes. This refined cryptanalysis leads to more precise algebraic analysis of cryptographic primitives based on algebraic S-boxes.

Considering the refined algebraic cryptanalysis, we propose a new one-way function, dubbed AIM, as an MPCitH-friendly symmetric primitive with high resistance to algebraic attacks. The security of AIM is comprehensively analyzed with respect to algebraic, statistical, quantum, and generic attacks. AIM is combined with the BN++ proof system, yielding a new signature scheme, dubbed AIMer. Our implementation shows that AIMer outperforms existing signature schemes based on symmetric primitives in terms of signature size and signing time.

Keywords: post-quantum, digital signature, MPC-in-the-head, algebraic analysis, Gröbner basis, power mapping

1 Introduction

With a substantial amount of research on quantum computers in recent years, the security threats posed by quantum computers are rapidly becoming a reality. Cryptography is considered particularly risky in the quantum computing environment since the security of most widely used public key schemes relies on the hardness of factoring or discrete logarithm, which is solved in polynomial time with a quantum computer [84].

* This work is submitted to Korean Post-Quantum Cryptography Competition (https://kpqc.or.kr).

** Both authors contributed equally to this work

† This work was done while B. Lee was a PhD student at KAIST.

‡ Jooyoung Lee was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) Grant funded by the Korea government (MSIT) (No. 2022-0-01202, Regional strategic industry convergence security core talent training business).

This encourages the cryptographic community to investigate post-quantum cryptographic schemes which are resilient to quantum attacks. NIST initiated a competition for post-quantum cryptography (PQC) standardization, and recently announced its selected algorithms: CRYSTALS-Kyber [82] as a public key encryption scheme, and CRYSTALS-Dilithium [72], Falcon [76], and SPHINCS$^+$ [53] as digital signature schemes.

MPC-in-the-Head based Signature. MPC-in-the-Head (MPCitH), proposed by Ishai et al. [54], is a paradigm to construct a zero-knowledge proof (ZKP) system from a multiparty computation (MPC) protocol. Its practicality is demonstrated by the ZKBoo scheme, the first efficient MPCitH-based proof scheme proposed by Giacomelli et al. [46]. One of the main applications of the MPCitH paradigm is to construct a post-quantum signature as follows. Given a one-way function f and an input-output pair (x, y) such that $f(x) = y$, one can construct a digital signature scheme with secret key x, public key y, and non-interactive zero-knowledge proof of the knowledge (NIZKPoK) of the secret x as a signature.

The main advantage of MPCitH-based signature schemes is that their security solely depends on the security of the one-way function used in key generation, which makes them more reliable compared to the schemes whose security is based on the hardness assumption of certain mathematical problems with a potential gap in the security reduction. For example, a multivariate signature scheme Rainbow [32] has been recently broken by exploiting the gap between its hardness assumption and the actual security [19]. Also, an isogeny-based key exchange algorithm SIKE [57] reveals its weakness as its security assumption does not hold [23]. In this context, MPCitH-based signature schemes are attracting significant attention as they provide diversity for the underlying hardness assumption. The recent call of NIST for additional digital signature schemes [75] also expressed primary interest in signature schemes that are not based on structured lattices.

Picnic [24] is the first and the most famous signature scheme based on the MPCitH paradigm; it combines an MPC-friendly block cipher LowMC [3] and an MPCitH proof system called ZKB++, which is an optimized variant of ZKBoo. Katz et al. [60] proposed a new proof system KKW by further improving the efficiency of ZKB++ with pre-processing, and updated Picnic accordingly. The updated version of Picnic was the only ZKP-based scheme that advanced to the third round of the NIST PQC competition.

LowMC is relatively a new design which can be computed efficiently in the MPC environment, where the AND operation is significantly expensive compared to XOR. There have been various attacks on LowMC, partially motivated by the LowMC challenge,[4] some of which have worked effectively [36, 37, 77, 10, 66, 11, 68, 69], and the LowMC parameters have been modified accordingly. Due to the security concern on LowMC, there have been attempts to construct MPCitH-based signature schemes from the one-wayness of the standard AES block cipher.[5] In this way, the hardness of key recovery from a single evaluation of AES is reduced to the security of the basing signature scheme. BBQ [79] and Banquet [15] are AES-based signature schemes, where BBQ employs the KKW proof system and Banquet improves BBQ by injecting shares for intermediate states.

To fully exploit efficient multiplication over a large field in the Banquet proof system, Dobraunig et al. proposed MPCitH-friendly ciphers LS-AES and Rain. They are substitution-permutation ciphers based on the inverse S-box over a large field [38]. This design strategy increases the efficiency of the resulting MPCitH-based signature scheme, while the number of rounds should be carefully determined by comprehensive analysis on any possible algebraic attack due to their simple algebraic structures. Kales and Zaverucha [58] proposed a number of optimization techniques to further improve the efficiency of the Baum and Nof's proof system [14], and their variant is called BN++. When Rain is combined with BN++, the resulting signature scheme enjoys the shortest signature size for the same level of signing/verification time (compared to existing MPCitH-based signatures) to the best of our knowledge.

Recently, a number of computational problems have been considered in the MPCitH paradigm such as the syndrome decoding problem [43], the subset sum problem [44], multivariate quadratic problems and a lattice problem [20, 18]. A PRF from alternating moduli has also been studied [35]. Most of these schemes are based on well-known security assumptions, while their performance is not competitive in general, so we regard them as rather orthogonal to our symmetric primitive-based scheme.

[4] https://lowmcchallenge.github.io/

[5] For a given plaintext-ciphertext pair, it is hard to find the secret key.

1.1 Our Contribution

The main contribution of this paper is two-fold. First, we refine algebraic cryptanalysis of power mapping based S-boxes over binary extension fields, and cryptographic primitives based on such S-boxes. In particular, we focus on the Gröbner basis and the XL (eXtended Linearization) attacks since they allow one to solve a system of equations from only a single evaluation of a one-way function, which is the case when it is used in an MPCitH-based signature scheme. Most of previous works on symmetric primitives over large fields analyzed their security against the Gröbner basis attack only over the large fields [1, 47, 4, 38]. Dobraunig et al. consider the analysis over $\mathbb{F}_2$ [38], but only deal with the equations of high degrees. We apply the Gröbner basis attack to the system of equations of low degrees over $\mathbb{F}_2$ using intermediate variables. When it comes to the Gröbner basis attack over $\mathbb{F}_2$, we experimentally show that the exact number of Boolean quadratic equations obtained from the underlying S-boxes is critical to correctly estimate the theoretic complexity based on the degree of regularity. Similarly, it turns out that the XL attack might be faster when all possible quadratic equations are found and used from the S-boxes. These results lead to more precise algebraic analysis of cryptographic primitives based on algebraic S-boxes.

Second, with a design rationale based on the refined algebraic cryptanalysis, we propose a new one-way function, dubbed AIM,[6] as an MPCitH-friendly symmetric primitive with high resistance to algebraic attacks. AIM uses Mersenne S-boxes based on power mappings with exponents of the form $2^e - 1$. Compared to the typical inverse S-box, Mersenne S-boxes turn out to provide higher resistance to algebraic attacks. The security of AIM is comprehensively analyzed with respect to algebraic, statistical, quantum and generic attacks. AIM is combined with the BN++ proof system, one of the state-of-the-art MPCitH proof systems working on large fields, yielding a new signature scheme, dubbed AIMer. The AIM function has been designed to fully exploit various optimization techniques of the BN++ proof system to reduce the overall signature size without significantly sacrificing the signing and the verification time.

We implement the AIMer signature scheme and compare its benchmark to existing post-quantum signatures on the same machine. Our implementation result is summarized in Section 6. Compared to the signature schemes based on the BN++ proof system combined with the 3-round (resp. 4-round) Rain, AIMer enjoys not only 8.21% (resp. 21.15%) shorter signature size but also 1.22% (resp. 13.41%) improved signing performance at 128-bit security level with the number of parties N being set to 16.

2 Preliminaries

2.1 Notation

For two vectors a and b over a finite field, their concatenation is denoted by $a\|b$. For a positive integer n, $\mathsf{hw}(n)$ denotes the Hamming weight of n in its binary representation, and we write $[n] = \{1, \cdots, n\}$.

Unless stated otherwise, all logarithms are to the base 2. The complexity of matrix multiplication of two $n \times n$ matrices is $O(n^\omega)$ for some ω such that $2 \leq \omega \leq 3$. The constant ω is called the matrix multiplication exponent, and it will be conservatively set to 2 in this paper.

For a set S, we will write $a \leftarrow S$ to denote that a is chosen uniformly at random from S. For a probability distribution $\mathcal{D}$, $a \leftarrow \mathcal{D}$ denotes that a is sampled according to the distribution $\mathcal{D}$.

In the multiparty computation setting, $x^{(i)}$ denotes the i-th party's additive share of x, which implies that $\sum_i x^{(i)} = x$.

2.2 Algebraic Attacks

An algebraic attack on a symmetric primitive is to model it as a system of multivariate polynomial equations and to solve it using algebraic techniques. A straightforward way of establishing a system of equations is to represent the output of the primitive as a polynomial of the input including the secret key. In order to

[6] This name is an abbreviation of Affine-Interleaved Mersenne S-boxes.

reduce the degree of the system of equations, intermediate variables might be introduced. For example, all the inputs and outputs of the underlying S-boxes can be regarded as independent variables.

One of the well-known methods of solving a system of equations is to define a system of linear equations by replacing every monomial of degree greater than one by a new variable and solve it, which is called *trivial linearization*. In the linearizaton, a large number of new variables might be introduced, and that many equations are also needed to determine a solution to the system of (linear) equations. However, in most ZKP-based digital signature schemes, one is given only a single evaluation of the underlying primitive, which limits the total number of equations thereof. For this reason, our focus will be put on algebraic attacks possibly using a small number of equations such as the Gröbner basis attack and the XL attack.

GRÖBNER BASIS ATTACK. The Gröbner basis attack is to solve a system of equations by computing its Gröbner basis. The attack consists of the following steps.

1. Compute a Gröbner basis in the *grevlex* (graded reverse lexicographic) order.
2. Change the order of terms to obtain a Gröbner basis in the *lex* (lexicographic) order.
3. Find a univariate polynomial in this basis and solve it.
4. Substitute this solution into the Gröbner basis and repeat Step 3.

When a system of equations has only finitely many solutions in its algebraic closure, its Gröbner basis in the *lex* order always contains a univariate polynomial. When a single variable of the polynomial is replaced by a concrete solution, the Gröbner basis still remains a Gröbner basis of the "reduced" system, allowing one to obtain a univariate polynomial again for the next variable. We refer to [81] for more details on Gröbner basis computation.

The complexity of Gröbner basis computation can be estimated using the *degree of regularity* of the system of equations [12]. Consider a system of m equations in n variables $\{f_i\}_{i=1}^m$. Let d_i denote the degree of f_i for $i = 1, 2, \dots, m$. If the system of equations is over-defined, i.e., $m > n$, then the degree of regularity can be estimated by the smallest of the degrees of the terms with non-positive coefficients for the following Hilbert series under the semi-regular assumption [45].

$$\mathrm{HS}(z) = \frac{1}{(1-z)^n} \prod_{i=1}^{m} (1 - z^{d_i}).$$

Given the degree of regularity d_{reg}, the complexity of computing a Gröbner basis of the system is known to be

$$O\left(\binom{n + d_{reg}}{d_{reg}}^{\omega}\right).$$

In the Gröbner basis attack, one always obtains an over-defined system of equations since each variable x should be contained in a finite field $\mathbb{F}_{p^e}$ for some characteristic p, and hence x satisfies $x^{p^e} - x = 0$ called a *field equation*. By including field equations in the system of equations, one can remove any possible solution outside $\mathbb{F}_{p^e}$ (in the algebraic closure). For some symmetric primitives, the field equations have not been taken into account in their analysis of the Gröbner basis attack [3, 4, 47, 38]. It does not mean that they are broken under the modified analysis, while considering the field equations would lead to more precise analysis of the Gröbner basis attack.

XL ATTACK. The XL algorithm, proposed by Courtois et al. [27], can be viewed as a generalization of the relinearization attack [62]. For a system of m quadratic equations in n variables over $\mathbb{F}_2$, the trivial linearization does not work if m is smaller than the number of monomials appearing in the system.

The XL algorithm extends the system of equations by multiplying all the monomials of degree at most $D-2$ for some $D > 2$ to obtain a larger number of linearly independent equations. Since the number of monomials of degree at most $D-2$ is $\sum_{i=1}^{D-2} \binom{n}{i}$, the resulting system consists of $(\sum_{i=0}^{D-2} \binom{n}{i})m$ equations of degree at most D with at most $\sum_{i=1}^{D} \binom{n}{i}$ monomials of degree at most D. When the number of equations equals the number of monomials as D grows, one can solve the extended system of equations by linearization.

The complexity of the XL attack depends on the number of linearly independent equations obtained from the XL algorithm, while we can loosely upper bound the number of linearly independent equations by

$(\sum_{i=0}^{D-2} \binom{n}{i})m$. Under the assumption that all the equations obtained from the XL algorithm are linearly independent, which is in favor of the attacker, we can search for the (smallest) degree D such that

$$\left(\sum_{i=0}^{D-2} \binom{n}{i}\right) m \geq T_D \tag{1}$$

where T_D denotes the exact number of monomials appearing in the extended system of equations, which is upper bounded by $\sum_{i=1}^{D} \binom{n}{i}$. Once D is fixed, the extended system of equations can be solved by trivial linearization whose time complexity is given as $O\left(T_D^{\omega}\right)$.

In the XL attack over $\mathbb{F}_2$, we do not contain the field equations since we already have taken advantage of them in counting the number of variables and extending the system. If we consider the field equations separately, monomials such as x^2 and x^3 should be regarded different ones from x for $x \in \mathbb{F}_2$, and they should be reduced to x by a linear combination with the field equation of x and its extended equations such as $x^2 + x = 0$ and $x^3 + x^2 = 0$. It would be better to regard them as the same as x without adding the field equations to the system.

2.3 BN++ Zero-knowledge Protocol

In this section, we briefly review the BN++ proof system [58], one of the state-of-the-art MPCitH zero-knowledge protocols. The BN++ protocol will be combined with our symmetric primitive AIM to construct the AIMer signature scheme which is fully described in Appendix E. At a high level, BN++ is a variant of the BN protocol [14] with several optimization techniques applied to reduce the signature size.

Protocol Overview. The BN++ protocol follows the MPCitH paradigm [54]. In order to check C multiplication triples $(x_j, y_j, z_j = x_j \cdot y_j)_{j=1}^{C}$ over a finite field $\mathbb{F}$ in the multiparty computation setting with N parties, *helping values* $((a_j, b_j)_{j=1}^{C}, c)$ are required, where $a_j \leftarrow \mathbb{F}, b_j = y_j$, and $c = \sum_{j=1}^{C} a_j \cdot b_j$. Each party holds secret shares of the multiplication triples $(x_j, y_j, z_j)_{j=1}^{C}$ and helping values $((a_j, b_j)_{j=1}^{C}, c)$. Then the protocol proceeds as follows.

- A prover is given random challenges $\epsilon_1, \cdots, \epsilon_C \in \mathbb{F}$.
- For $i \in [N]$, the i-th party locally sets $\alpha_1^{(i)}, \cdots, \alpha_C^{(i)}$ where $\alpha_j^{(i)} = \epsilon_j \cdot x_j^{(i)} + a_j^{(i)}$.
- The parties open $\alpha_1, \cdots, \alpha_C$ by broadcasting their shares.
- For $i \in [N]$, the i-th party locally sets

$$v^{(i)} = \sum_{j=1}^{C} \epsilon_j \cdot z_j^{(i)} - \sum_{j=1}^{C} \alpha_j \cdot b_j^{(i)} + c^{(i)}.$$

- The parties open v by broadcasting their shares and output `Accept` if $v = 0$.

The probability that there exist incorrect triples and the parties output `Accept` in a single run of the above steps is upper bounded by $1/|\mathbb{F}|$.

Signature Size. By applying the Fiat-Shamir transform [39], one can obtain a signature scheme from the BN++ proof system. In this signature scheme, the signature size is given as

$$6\lambda + \tau \cdot (3\lambda + \lambda \cdot \lceil \log_2(N) \rceil + \mathcal{M}(C)),$$

where λ is the security parameter, C is the number of multiplication gates in the underlying symmetric primitive, and $\mathcal{M}(C) = (2C + 1) \cdot \log_2(|\mathbb{F}|)$. In particular, $\mathcal{M}(C)$ has been defined so from the observation that sharing the secret share offsets for $(z_j)_{j=1}^{C}$ and c, and opening shares for $(\alpha_j)_{j=1}^{C}$ occurs for each repetition, using C, 1, and C elements of $\mathbb{F}$, respectively. For more details, we refer to [58].

Optimization Techniques. If multiplication triples use an identical multiplier in common, for example, given (x_1, y, z_1) and (x_2, y, z_2), then the corresponding α values can be batched to reduce the signature size.

Instead of computing $\alpha_1 = \epsilon_1 \cdot x_1 + a_1$ and $\alpha_2 = \epsilon_2 \cdot x_2 + a_2$, $\alpha = \epsilon_1 \cdot x_1 + \epsilon_2 \cdot x_2 + a$ is computed, and v is defined as

$$v = \epsilon_1 \cdot z_1 + \epsilon_2 \cdot z_2 - \alpha \cdot y + c,$$

where $c = a \cdot y$. This technique is called *repeated multiplier* technique. Our symmetric primitive design allows us to take full advantage of this technique to reduce the number of α values in each repetition of the protocol.

If the output of the multiplication z_i can be locally generated from each share, then the secret share offset is not necessarily included in the signature.

3 Refining Algebraic Cryptanalysis of Power Functions over Binary Fields

Representation in $\mathbb{F}_2$ and its Extension Field. When a symmetric primitive is defined with arithmetic in a large field, it is straightforward to establish a system of equations from a single evaluation of the primitive using the same field arithmetic. If the underlying field is a binary extension field $\mathbb{F}_{2^n}$ for some n, then it is also possible to establish a system of equations over $\mathbb{F}_2$. Suppose that $\{1, \beta, \ldots, \beta^{n-1}\}$ is a basis of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$. Then each variable $x \in \mathbb{F}_{2^n}$ can be represented as n Boolean variables $x_0, x_1, \ldots, x_{n-1} \in \mathbb{F}_2$ by setting $x = \sum_{i=0}^{n-1} x_i \beta^i$. Using the representation of β^n with respect to this basis, every polynomial equation over $\mathbb{F}_{2^n}$ can be transformed into n equations over $\mathbb{F}_2$.

On the other hand, a linear equation over $\mathbb{F}_2$ is represented by a *linearized polynomial* over $\mathbb{F}_{2^n}$:

$$\sum_{i=0}^{n-1} a_i x^{2^i} = a_0 x + a_1 x^{2^1} + a_2 x^{2^2} + \cdots + a_{n-1} x^{2^{n-1}} \tag{2}$$

where $a_0, a_1, \ldots, a_{n-1} \in \mathbb{F}_{2^n}$.

Suppose that variables x and y in $\mathbb{F}_{2^n}$ are represented by $\{x_i\}_{i=0}^{n-1}$ and $\{y_i\}_{i=0}^{n-1}$, respectively, in $\mathbb{F}_2$. If $y = x^a$ for some a, then each y_i is represented as a multivariate polynomial of x_i's of degree $\mathsf{hw}(a)$. For instance, the inverse S-box $y = x^{2^n-2}$ can be represented as a system of n Boolean equations of degree $n-1$.

For most of the previous works on symmetric primitives over large fields, their security against algebraic attacks has been analyzed mainly over the large fields [1, 47, 4, 38]. On the other hand, it is also possible to represent the primitive by a system of equations over the prime subfield. For example, Rain [38] has been analyzed by representing its inverse S-box by a system of Boolean equations of degree $n-1$ induced from the equation $y = x^{2^n-2}$ over $\mathbb{F}_{2^n}$. However, we note that it is also possible to build a quadratic system of equations over $\mathbb{F}_2$ using the equation $xy = 1$ over $\mathbb{F}_{2^n}$. In this work, we will consider all possible representations of the underlying algebraic S-boxes, and analyze the security of AIM against the Gröbner basis attack and the XL attack.

Number of Quadratic Equations. The efficiency of algebraic cryptanalysis heavily depends on the number of variables, the number of equations, and their degrees for the system of equations. As discussed above, a powering function $y = x^a$ over $\mathbb{F}_{2^n}$ can be represented as a system of n equations of degree $\mathsf{hw}(a)$ over $\mathbb{F}_2$. The resulting equations are explicit ones in a sense that each output variable is represented by an equation only in the input variables. However, their *implicit* representation might consist of equations of degrees smaller than the explicit ones. For example, $y = x^{2^n-2}$ obtained from the inverse S-box is equivalent to the quadratic equation $xy = 1$ over $\mathbb{F}_{2^n}$, assuming the input x is nonzero, or a certain set of n quadratic equations in n variables over $\mathbb{F}_2$.

Implicit representation over $\mathbb{F}_2$ might also increase the number of (linearly independent) equations. There has been a significant amount of research on the number of linearly independent quadratic equations obtained from power functions over $\mathbb{F}_{2^n}$ [26, 28, 74, 50]. For example, Cheon and Lee [26] showed that one has $5n$ quadratic equations over $\mathbb{F}_2$ from $xy = 1$ over $\mathbb{F}_{2^n}$. Later, Courtois et al. [28] pointed out that the relation $xy = 1$ holds only when x and y are nonzero, and further showed that $5n-1$ linearly independent quadratic equations are obtained from the exact representation of the inverse S-box. In the following, we will study how the number of quadratic equations obtained from a power mapping based S-box affects the complexity of the Gröbner basis attack and the XL attack on a symmetric primitive based on the S-box. As a concrete primitive, we consider a single key variant of the Even-Mansour cipher [41], where the inner permutation is instantiated as $R \circ S \circ L$ for random affine maps L and R, and the target S-box S.

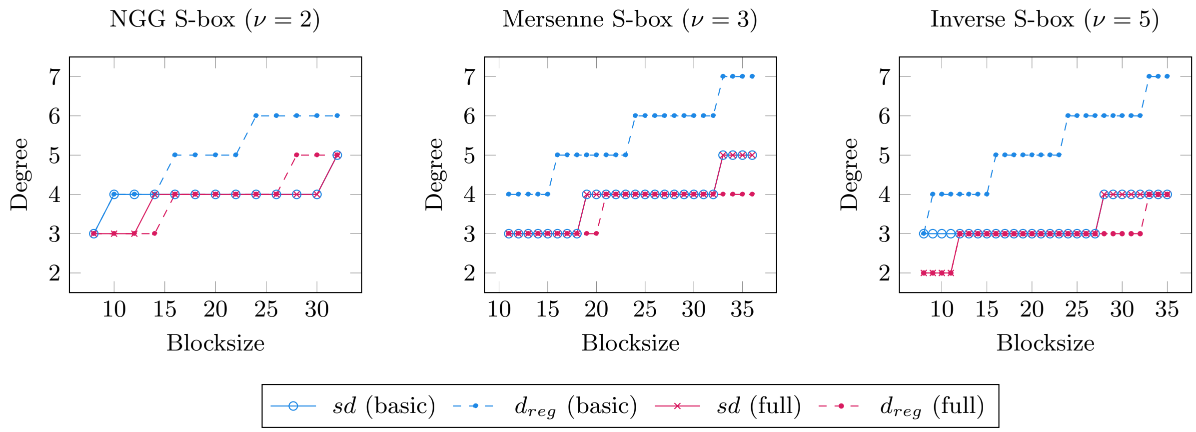

Fig. 1: The estimated degree of regularity d_{reg} and the solving degree sd for the basic and the full systems of equations constructed from a single evaluation of an Even-Mansour cipher built on top of each of NGG, Mersenne, and the inverse S-boxes, where each S-box generates νn linearly independent quadratic equations.

3.1 Gröbner Basis Attack over $\mathbb{F}_2$

n	Degree of Regularity					Complexity (bits)				
	$\nu=1$	$\nu=2$	$\nu=3$	$\nu=4$	$\nu=5$	$\nu=1$	$\nu=2$	$\nu=3$	$\nu=4$	$\nu=5$
128	17	11	9	8	7	144.6	104.9	90.1	82.2	74.0
192	23	15	12	10	9	204.0	148.8	125.5	108.9	100.3
256	29	19	14	12	10	263.1	192.6	152.5	135.2	117.0

Table 1: The estimated degree of regularity for an Even-Mansour cipher and the corresponding time complexity for computing a Gröbner basis according to the value of ν and the block size $n \in \{128, 192, 256\}$, where each S-box generates νn linearly independent quadratic equations.

In order to see how the number of quadratic equations from a power mapping based S-box affects the time complexity of the Gröbner basis computation, we compare the theoretic estimation of the degree of regularity and the *solving degree* [33], which is the highest degree reached during the actual Gröbner basis computation, for toy parameters. The solving degrees are obtained with *grevlex* order.

We will consider three types of S-boxes: the inverse S-box $y = x^{2^n-2}$, a Mersenne S-box $y = x^{2^e-1}$ for some e, and an S-box represented by $y = x^{2^{s+1}+2^{s-1}-1}$ for $n = 2s$, which is called an *NGG* S-box (after the authors of [74] that studied its properties). Each S-box is a powering function of the form $y = x^a$ where $\mathsf{hw}(a+1) \in \{0, 1, 2\}$. Since x^{a+1} is of degree at most two over $\mathbb{F}_2$, each S-box defines n quadratic equations over $\mathbb{F}_2$ from an implicit equation $xy = x^{a+1}$.

It is known that an NGG S-box defines $2n$ quadratic equations over $\mathbb{F}_2$ if $n \geq 8$ [74]. For a Mersenne S-box, we find e such that $y = x^{2^e-1}$ defines $3n$ quadratic equations over $\mathbb{F}_2$ using the algorithm proposed in

[74]. When it comes to the inverse S-box, we will assume that it defines $5n$ quadratic equations over $\mathbb{F}_2$ from the quadratic relation $xy = 1$ over $\mathbb{F}_{2^n}$ [26].[7] Denoting the number of quadratic equations of the S-boxes by νn, we have $\nu = 2$ for the NGG S-box, $\nu = 3$ for the Mersenne S-box, and $\nu = 5$ for the inverse S-box.

For each S-box, we consider two different types of systems of equations: the *basic* system containing only n quadratic equations directly obtained from the implicit quadratic relation such as $xy = 1$ and $xy = x^{2^e}$, and the *full* system containing the exact number of quadratic equations induced from the S-box definition. The exact quadratic equations describing the full system is computed by the algorithm proposed in [50].

With toy parameters, we compute a Gröbner basis for a system of equations defined by a single evaluation of an Even-Mansour ciphers based on each of the three S-boxes, using MAGMA [22]. Figure 1 compares the estimated degree of regularity and the solving degree *sd*. We observe that for both the basic and the full systems, their solving degrees are close to the theoretically estimated degree of regularity for the full system.

The three S-boxes differ in the actual running time of Gröbner basis computation as shown in Appendix A. We observe that Gröbner basis computation becomes faster given a larger number of quadratic equations.

In Table 1, we summarize the estimated degree of regularity for an evaluation of an Even-Mansour cipher, and the corresponding time complexity for Gröbner basis computation for various values of ν and $n \in \{128, 192, 256\}$. We observe that the time complexity significantly decreases as ν grows. We can conclude that the exact number of quadratic equations from an S-box, represented by the constant ν, is critical to algebraic cryptanalysis of a primitive built on the S-box.

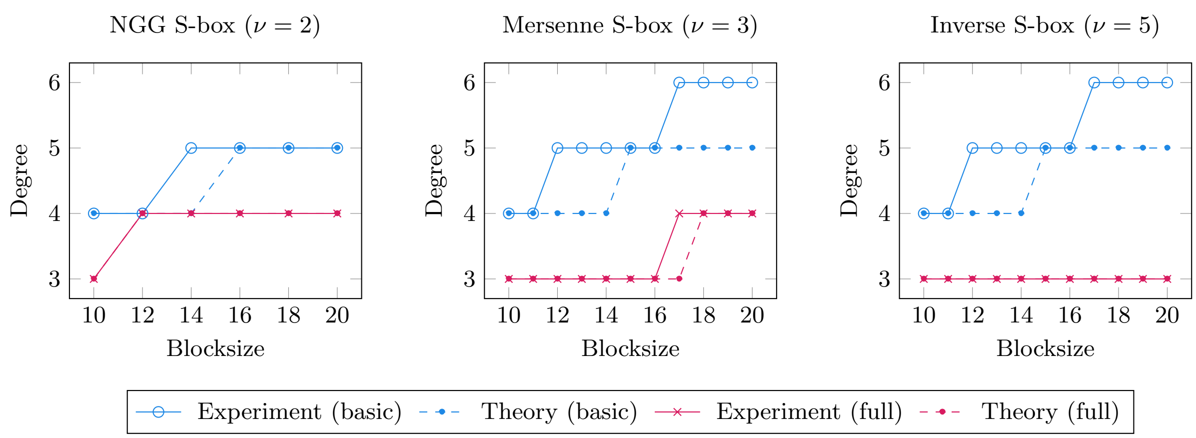

Fig. 2: Theoretical and experimental estimations for the target degree D of the XL attack for the basic and the full systems of equations constructed from a single evaluation of an Even-Mansour ciphers based on NGG, Mersenne, and the inverse S-boxes, where each S-box generates νn linearly independent quadratic equations.

3.2 XL Attack over $\mathbb{F}_2$

To see the impact of the number of quadratic equations from the underlying S-box on the XL attack, we implemented the XL algorithm for an Even-Mansour cipher with toy parameters. We observed that all the systems induced by NGG, Mersenne, and the inverse S-boxes are dense; all the monomials of degrees up to D appear during the execution of the algorithm.

Figure 2 compares the smallest degree D satisfying (1) (denoted "theory") and the actual degree D for which the number of extended equations is almost the same as the number of monomials (denoted

[7] As aforementioned, the inverse S-box defines $5n - 1$ quadratic equations [28]. However, when used in ZKP-based signature schemes, it is assumed that $xy = 1$ for nonzero x and y [15, 38].

"experiment"). There is no significant difference between the theoretical and experimental estimations, while we observe that a smaller target degree D is obtained with the full system. We also observe that the gap of the target degrees between the basic and the full systems becomes larger for a larger parameter ν.

4 AIM: Our New Symmetric Primitive

4.1 Specification

AIM is designed to be a "tweakable" one-way function so that it offers multi-target one-wayness. Given input/output size n and an $(\ell+1)$-tuple of exponents $(e_1, \ldots, e_\ell, e_*) \in \mathbb{Z}^{\ell+1}$, $\mathsf{AIM} : \{0,1\}^n \times \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}$ is defined by

$$\mathsf{AIM}(\mathsf{iv}, \mathsf{pt}) = \mathsf{Mer}[e_*] \circ \mathsf{Lin}[\mathsf{iv}] \circ \mathsf{Mer}[e_1, \ldots, e_\ell](\mathsf{pt}) \oplus \mathsf{pt}$$

where each function will be described below. See Figure 3 for the pictorial description of AIM with $\ell = 3$.

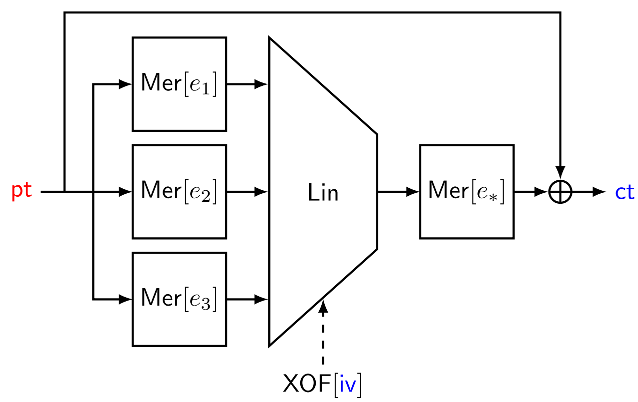

Fig. 3: The AIM-V one-way function with $\ell = 3$. The input pt (in red) is the secret key of the signature scheme, and $(\mathsf{iv}, \mathsf{ct})$ (in blue) is the corresponding public key.

S-boxes. In AIM, S-boxes are exponentiation by Mersenne numbers over a large field. More precisely, for $x \in \mathbb{F}_{2^n}$,

$$\mathsf{Mer}[e](x) = x^{2^e - 1}$$

for some e. Note that this map is a permutation if $\gcd(e, n) = 1$. As an extension, $\mathsf{Mer}[e_1, \ldots, e_\ell] : \mathbb{F}_{2^n} \to \mathbb{F}_{2^n}^\ell$ is defined by

$$\mathsf{Mer}[e_1, \ldots, e_\ell](x) = \mathsf{Mer}[e_1](x) \| \ldots \| \mathsf{Mer}[e_\ell](x).$$

Linear Components. AIM includes two types of linear components: an affine layer and feed-forward. The affine layer is multiplication by an $n \times \ell n$ random binary matrix A_{iv} and addition by a random constant $b_{\mathsf{iv}} \in \mathbb{F}_2^n$. The matrix

$$A_{\mathsf{iv}} = \left[A_{\mathsf{iv},1} \middle| \ldots \middle| A_{\mathsf{iv},\ell}\right] \in (\mathbb{F}_2^{n \times n})^\ell$$

is composed of ℓ random invertible matrices $A_{\mathsf{iv},i}$. The matrix A_{iv} and the vector b_{iv} are generated by an extendable output function (XOF) with the initial vector iv.[8] Each matrix $A_{\mathsf{iv},i}$ can be equivalently represented

[8] The invertible matrices can be generated by various ways. (e.g., rejection sampling, generating LU decomposition, etc.)

by a linearized polynomial $L_{\mathsf{iv},i}$ on $\mathbb{F}_{2^n}$. For $x = (x_1, \ldots, x_\ell) \in (\mathbb{F}_{2^n})^\ell$,

$$\mathsf{Lin}[\mathsf{iv}](x) = \sum_{1 \le i \le \ell} L_{\mathsf{iv},i}(x_i) \oplus b_{\mathsf{iv}}.$$

By abuse of notation, we will denote Ax as the same meaning as $\sum_{1\le i\le \ell} L_{\mathsf{iv},i}(x_i)$. Feed-forward operation, which is addition by the input itself, makes the entire function non-invertible.

Recommended Parameters. Recommended sets of parameters for $\lambda \in \{128, 192, 256\}$-bit security are given in Table 2. The number of S-boxes is determined by taking into account the complexity of the XL attack, which is described in Section 5.2. Exponents e_1 and e_* are chosen as small numbers to provide smaller differential probability, and exponent e_2 is chosen so that $e_2 \cdot e_* \ge \lambda$, while all the exponents are distinct in the same set of parameters. The irreducible polynomials for extension fields $\mathbb{F}_{2^{128}}$, $\mathbb{F}_{2^{192}}$, and $\mathbb{F}_{2^{256}}$ are the same as those used in Rain [38].

Scheme	λ	n	ℓ	e_1	e_2	e_3	e_*
AIM-I	128	128	2	3	27	-	5
AIM-III	192	192	2	5	29	-	7
AIM-V	256	256	3	3	53	7	5

Table 2: Recommended sets of parameters of AIM.

4.2 Design Rationale

Choice of Field. When a symmetric primitive is built upon field operations, the field is typically binary since bitwise operations are cheap in most of modern architectures. However, when the multiplicative complexity of the primitive becomes a more important metric for efficiency, it is hard to generally specify which type of field has merits with respect to security and efficiency.

Focusing on a primitive for MPCitH-style zero-knowledge protocols, a primitive over a large field generally requires a small number of multiplications, leading to shorter signatures. However, any primitive operating on a large field of large prime characteristic might permit algebraic attacks since the number of variables and the algebraic degree will be significantly limited for efficiency reasons. On the other hand, binary extension fields enjoy both advantages from small and large fields. In particular, matrix multiplication is represented by a polynomial of high algebraic degree without increasing the proof size.

Algebraically Sound S-boxes. In an MPCitH-style zero-knowledge protocol, the proof size of a circuit is usually proportional to the number of nonlinear operations in the circuit. In order to minimize the number of multiplications, one might introduce intermediate variables for some wires of the circuit. For example, the inverse S-box ($S(x) = x^{-1}$) has high (bitwise) algebraic degree $n-1$, while it can be simply represented by a quadratic equation $xy = 1$ by letting the output from the S-box be a new variable y. When an S-box is represented by a quadratic equation of its input and output, we will say it is *implicitly quadratic*. In particular, we consider implicitly quadratic S-boxes which are represented by a single multiplication over $\mathbb{F}_{2^n}$. This feature makes the proof size short and mitigates algebraic attacks at the same time.

The inverse S-box is one of the well-studied implicitly quadratic S-boxes. The inverse S-box has been widely adopted to symmetric ciphers [29, 6, 83] due to its nice cryptographic properties. It is invertible, is of high-degree, has good enough differential uniformity and nonlinearity. Recently, it is used in symmetric primitives for advanced cryptographic protocols such as multiparty computation and zero-knowledge proof [47, 48, 38].

Meanwhile, the inverse S-box has one minor weakness; a single evaluation of the n-bit inverse S-box as a form of $xy = 1$ produces $5n - 1$ linearly independent quadratic equations over $\mathbb{F}_2$ [28]. The complexity of an algebraic attack is typically bounded (with heuristics) by the degree and the number of equations, and the number of variables. In particular, an algebraic attack is more efficient with a larger number of equations, while this aspect has not been fully considered in the design of recent symmetric ciphers based on algebraic S-boxes. When the number of rounds is small, this issue might be critical to the overall security of the cipher. For more details, see Section 5.2.

With the above observation, we tried to find an invertible S-box of high-degree which is moderately resistant to differential/linear cryptanalysis as well as implicitly quadratic, and *producing only a small number of quadratic equations*. Since our attack model does not allow multiple queries to a single instance of AIM, we allow a relaxed condition on the DC/LC resistance, not being necessarily maximal. As a family of S-boxes that beautifully fit all the conditions, we choose a family of Mersenne S-boxes; they are exponentiation by Mersenne numbers $2^e - 1$ such that $\gcd(n, e) = 1$, are invertible, are of high-degree, need only one multiplication for its proof, produce only $3n$ Boolean quadratic equations with its input and output, and provide moderate DC/LC resistance. Furthermore, when the implicit equation $xy = x^{2^e}$ of a Mersenne S-box is computed in the BN++ proof system, it is not required to broadcast the output share since the output of multiplication x^{2^e} can be locally computed from the share of x.

Repetitive Structure. The efficiency of the BN++ proof system partially comes from the optimization technique using *repeated multipliers*. When a multiplier is repeated in multiple equations to prove, the proof can be done in a batched way, reducing the overall signature size. In order to maximize the advantage of repeated multipliers, we put S-boxes in parallel at the first round in order to make the S-box inputs the same. Then, we put only one S-box at the second round with feed-forward. In this way, all the implicit equations have the same multiplier.

Affine Layer Generation. The main advantage of using binary affine layers in large S-box-based constructions is to increase the algebraic degree of equations over the large field. Multiplication by a random $n \times n$ binary matrix can be represented as (2). Similarly, our design uses a random affine map from $\mathbb{F}_2^{\ell n}$ to $\mathbb{F}_2^n$. In order to mitigate multi-target attacks (in the multi-user setting), the affine map is uniquely generated for each user; each user's iv is fed to an XOF, generating the corresponding linear layer.

5 Security Analysis

In order for the signature scheme to be secure, AIM with fixed iv should be preimage-resistant. An adversary is given a randomly chosen iv and an output ct that is defined by iv and a randomly chosen input pt^*. Given such a pair $(\mathsf{iv}, \mathsf{ct})$, the adversarial goal is to find any pt (not necessarily the same as pt^*) such that $\mathsf{AIM}(\mathsf{iv}, \mathsf{pt}) = \mathsf{ct}$. In the multi-user setting, the adversary is given multiple IV-output pairs $\{(\mathsf{iv}_i, \mathsf{ct}_i)\}_i$, and tries to find any pt such that $\mathsf{AIM}(\mathsf{iv}_i, \mathsf{pt}) = \mathsf{ct}_i$ for some i.

5.1 Brute-force Attack

Although a brute-force attack on a symmetric primitive is rather trivial (compared to public key cryptosystems), we estimate its gate-count complexity to compare the concrete security of AIM and AES.

Each S-box $\mathsf{Mer}[e]$ requires $2(e-1)$ multiplications over $\mathbb{F}_{2^n}$. Assuming that a single $\mathbb{F}_{2^n}$-multiplication requires n^2 AND gates and n^2 XOR gates, the gate-count complexity of a brute-force attack is given as 2^{149}, $2^{214.4}$, and 2^{280} for AIM-I, AIM-III, and AIM-V, respectively. It implies that a brute-force attack on AIM is more costly than AES for each category of security strength.

5.2 Algebraic Attacks

Since our attack model does not allow multiple evaluations for a single instance of AIM, we do not consider interpolation, higher-order differential, and cube attacks. As mentioned in Section 3, we mainly consider the Gröbner basis attack and the XL attack using a single evaluation of AIM.

How to Build Systems of Equations from AIM. AIM can be represented as a system of polynomial equations over either $\mathbb{F}_{2^n}$ or $\mathbb{F}_2$. For the former case, since the random affine layer is of degree 2^{n-1} over $\mathbb{F}_{2^n}$ with high probability, it is infeasible to solve such systems no matter how the system of equations is built. For the latter case, we need to consider multiple ways of building a system of polynomial equations.

Let x and y_i denote the input of AIM (namely, pt) and the output of $\mathsf{Mer}[e_i]$, $i = 1, \ldots, \ell$, respectively. Then, for AIM-V with $\ell = 3$, one might want to solve the following system of quadratic equations in x, y_1, y_2, and y_3:

$$\begin{aligned} x \cdot y_1 &= x^{2^{e_1}}, \\ x \cdot y_2 &= x^{2^{e_2}}, \\ x \cdot y_3 &= x^{2^{e_3}}, \\ \mathsf{Lin}(y_1, y_2, y_3) \cdot (x \oplus \mathsf{ct}) &= \mathsf{Lin}(y_1, y_2, y_3)^{2^{e_*}}. \end{aligned} \tag{3}$$

This system consists of $4n$ quadratic equations in $4n$ variables. As mentioned in Section 3, the relation between the input and the output of each n-bit Mersenne S-box produces $3n$ Boolean quadratic equations, so this system can be extended to $12n$ quadratic equations. This is the only way of obtaining quadratic equations from a single evaluation of AIM-V. For AIM-I and AIM-III with $\ell = 2$, one obtains $9n$ quadratic equations in $3n$ variables with a smaller number of S-boxes. Since all the equations in this system is quadratic, the complexity to solve this system is quite competitive.

For AIM-V, one might also establish a system of equations in x and y_2 as follows.

$$\begin{aligned} & x \cdot y_2 = x^{2^{e_2}}, \\ & \mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, \mathsf{Mer}[e_3](x)) \cdot (x \oplus \mathsf{ct}) \\ & = \mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, \mathsf{Mer}[e_3](x))^{2^{e_*}}. \end{aligned} \tag{4}$$

Considering the constant ν, this system consists of $3n$ equations of degree 2, and $3n$ equations of degree $\max(e_1, e_3) + 1$, all in $2n$ variables. For AIM-I and AIM-III, one also obtains a similar type of system of equations by ignoring y_3 in the above system of equations. All the possible systems of equations can be found in Appendix B.

Gröbner Basis Attack. For AIM, the system of equations of type (4) turns out to permit the most efficient computation of a Gröbner basis; the corresponding Hilbert series is given as

$$\frac{(1 - z^2)^{5n}(1 - z^d)^{3n}}{(1 - z)^{2n}}$$

including the field equations, where $d = \max(e_1, e_3) + 1$. The estimated degrees of regularity and the corresponding time complexities of computing Gröbner bases are given in Table 3 for AIM-I, III, V.

XL Attack. For AIM-I and III, the system of quadratic equations of type (3) permits the most efficient XL-attack. Assuming $T_D = \sum_{i=1}^{D} \binom{3n}{i}$,[9] one can find the smallest degree D satisfying (1).

The XL attack on AIM-V is the most efficient with the following system of equations: for variables x, y_2, and y_3,

$$\begin{aligned} x \cdot y_2 &= x^{2^{e_2}}, \\ x \cdot y_3 &= x^{2^{e_3}}, \\ \mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, y_3) \cdot (x \oplus \mathsf{ct}) &= \mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, y_3)^{2^{e_*}}. \end{aligned} \tag{5}$$

[9] As mentioned in Section 3.2, we observe that the system of quadratic equations defined by a single evaluation of an Even-Mansour cipher combined with the Mersenne S-box is dense for toy parameters. The same result is obtained from the quadratic system of AIM.

Considering the constant ν, this system consists of $6n$ quadratic equations, and $3n$ equations of degree 4, all in $3n$ variables. Since this system contains equations of different degrees, the target degree D should be determined in a slightly different way. Suppose that we raise the degree of the above system up to D, where $D \geq 4$. In order for the number of equations in the extended system to be at least the number of monomials, D should satisfy

$$6n \cdot \sum_{j=0}^{D-2} \binom{3n}{j} + 3n \cdot \sum_{j=0}^{D-4} \binom{3n}{j} \geq T_D.$$

Assuming that all possible monomials of degree at most D appear in the extended system of equations, namely, $T_D = \sum_{i=1}^{D} \binom{3n}{i}$, one can find the smallest D satisfying the above inequality.

With the *optimal* systems of AIM-I, III, V, the target degrees D and the corresponding time complexities of the XL attack are given in Table 3. We note that the time complexity of the XL attack has been estimated under the strong assumption that all the equations obtained by the XL algorithm are linearly independent, which might not be the case in general. Even with this strong assumption, we see that AIM is secure against the XL attack for all the parameter sets.

Scheme	n	ν	Gröbner Basis Attack			XL Attack		
			System	d_{reg}	Time (bits)	System	D	Time (bits)
AIM-I	128		(4)	22	214.9	(3)	12	148.0
AIM-III	192	3	(4)	31	310.6	(3)	15	194.1
AIM-V	256		(4)	40	406.2	(5)	20	260.6
	128		Quadratic	14	168.5	Quadratic	10	$127.9^\dagger$
Rain$_3$	192	5	Quadratic	19	235.9	Quadratic	12	$162.1^\dagger$
	256		Quadratic	24	303.1	Quadratic	13	$183.9^\dagger$
	128		Quadratic	17	219.2	Quadratic	11	147.3
Rain$_4$	192	5	Quadratic	24	303.1	Quadratic	13	$183.9^\dagger$
	256		Quadratic	30	385.9	Quadratic	15	$219.2^\dagger$

Table 3: Security against the Gröbner basis attack and the XL attack for AIM and Rain. d_{reg} is the estimated degree of regularity and D is the target degree of the XL attack on the system. †The complexity of the XL attack has been estimated based on the strong *independence* assumption with $\omega = 2$, so these values do not imply that Rain has been broken.

AIM vs. Rain. Table 3 compares the theoretical complexities of the Gröbner basis attack and the XL attack on AIM and Rain. The time complexities have been estimated using the most efficient systems from the adversarial point of view. For AIM, the system of quadratic equations is not the most efficient to mount algebraic attacks. When it comes to Rain, the quadratic representation of the inverse S-box, namely $xy = 1$, leads to the most efficient system of quadratic equations to mount algebraic attacks. Compared to Rain, AIM provides stronger security against the Gröbner basis and the XL attacks.

For the systems of quadratic equations from Rain$_3$ (resp. Rain$_4$) and AIM with $\ell = 2$ (resp. $\ell = 3$), the number of variables and the number of *basic* equations are the same. The only difference lies in the constant ν, indicating how many linearly independent quadratic equations are derived from the basic equations. In Appendix C, we perform Gröbner basis computation on the quadratic systems of AIM and Rain with toy parameters to justify the theoretical impact of ν.

Compared to Rain, the algebraic representation of the Mersenne S-boxes in AIM allows one to establish many different systems of "moderate-degree" polynomial equations. That said, AIM enjoys better security than Rain thanks to the smaller value of ν of the underlying S-boxes.

Algebraic Attacks on Symmetric Primitives with Large S-box. Besides Rain, several symmetric primitives based on large fields have been proposed with applications to zero-knowledge proof systems such as MiMC [1], Starkad/Poseidon [47], and Jarvis [8]. Some of them have been analyzed with algebraic attacks exploiting the property that their linear layers are represented as polynomials of low degrees over large fields [2, 40]. However, AIM uses a randomized linear layer which is expected to have degree 2^{n-1} over $\mathbb{F}_{2^n}$. For this reason, the above attacks would not apply to AIM.

Applicability of Algebraic Attacks on LowMC. LowMC [3] is the first FHE/MPC-friendly block cipher, and one of its applications is to the Picnic signature scheme. LowMC has been analyzed in the context of the signature scheme, where an adversary is given only a single plaintext-ciphertext pair. In this setting, a number of algebraic attacks have been proposed [10, 11, 67, 34, 68, 9], mainly based on two algebraic techniques: linearization by guessing, and the polynomial method [16].

The main idea of linearization-based algebraic attacks on LowMC, first proposed in [10], is to linearize the underlying S-boxes by guessing a single output bit for each S-box evaluation. In this way, one obtains a system of low-degree polynomial equations at the cost of guessing a small number of bits, and it can be solved efficiently. This linearization technique has been further extended [11, 67]. However, this type of attacks work only when the underlying S-boxes are of small size. When it comes to AIM, its large S-boxes yield dense implicit equations over $\mathbb{F}_2$, which makes the *guess-and-linearization* infeasible.

The polynomial method [16] has been studied in complexity theory, and later found its application to the design of algorithms for certain problems [86], one of which is to solve a system of polynomial equations over a finite field. The resulting algorithm is known as the first algorithm that achieves exponential speedup over the exhaustive search even in the worst case [70]. Recently, Dinur [34] proposed a generic equation-solving algorithm based on the polynomial method with time complexity $O(n^2 \cdot 2^{(1-1/(2.7d))n})$ where n is the number of variables and d is the degree of the system. One arguable issue of this algorithm is its high memory complexity of $O(n^2 \cdot 2^{(1-1/(1.35d))n})$, making it infeasible in practice. For AIM, the memory complexity required by Dinur's algorithm exceeds the security level, i.e., more than 2^λ bits of memory is required for each level of security λ (see Table 9 in Appendix B for the details). Subsequent works [68, 9] proposed to reduce the memory complexity of the algorithm at the cost of slightly increased time complexity, while these variants do not apply to AIM since they all follow the guess-and-linearization strategy.

5.3 Quantum Attacks

Quantum attacks are classified into two types according to the attack model. In the Q1 model, an adversary is allowed to use quantum computation without making any quantum query, while in the Q2 model, both quantum computation and quantum queries are allowed [88].

As a generic algorithm for exhaustive key search, Grover's algorithm has been known to give quadratic speedup compared to the classical brute-force attack [49]. In this section, we investigate if any specialized quantum algorithm targeted at AIM might possibly achieve better efficiency than Grover's algorithm in the Q1 model.

Cost of Grover's Algorithm. We consider the cost metric of NIST [75], which is defined as the product of the quantum circuit size and the quantum circuit depth with respect to Clifford and T gates.

Given a one-way function f taking n bits as input, the circuit size and the depth of the preimage-finding attack on f using Grover's algorithm is estimated as follows [55].

$$\text{(Grover's circuit size/depth)} = \text{(size/depth of } f) \times 2 \times \left\lfloor \frac{\pi}{4}\sqrt{2^n} \right\rfloor .$$

The quantum circuit size and the depth of AIM can be computed in a modular manner. AIM is based on three types of operations: finite field multiplication, finite field squaring, and random matrix multiplication. The cost of finite field multiplication is estimated based on the state-of-the-art result of Toffoli-depth one implementation of finite field multiplication [56], while we ignore the cost of finite field squaring since it is far more efficient than finite field multiplication [73]. For random matrix multiplication and Toffoli gate decomposition, we refer to the recent implementation of LowMC [55] and the implementation of [5] (using 8 Clifford gates and 7 T gates with depth 8), respectively.

Table 4 summarizes the total number of operations and the number of operations executed in serial (depth) for each type of operation. Based on these numbers and the above references, the total cost of Grover's algorithm on AIM is also given (in log) for each level of security. We see that AIM-I, AIM-III and AIM-V satisfy the security level I, III and V, respectively.[10]

Scheme	#Operations, Depth			Total Cost	Level of Security
	FF Mul	FF Square	Mat Mul		
AIM-I	32, 30	32, 30	1, 1	161.2	I ($\geq$157)
AIM-III	38, 34	38, 34	1, 1	227.7	III ($\geq$221)
AIM-V	64, 56	64, 56	1, 1	292.9	V ($\geq$285)

Table 4: The number of operations and the depth for each type of operation used in AIM, and the total cost of Grover's algorithm on AIM for each level of security.

Quantum Algebraic Attack. When an algebraic root-finding algorithm works over a small field, the guess-and-determine strategy might be effectively combined with Grover's algorithm, reducing the overall time complexity.

The GroverXL algorithm [17] is a quantum version of the FXL algorithm [27], which solves a system of multivariate quadratic equations over a finite field. A single evaluation of AIM can be represented by Boolean quadratic equations using intermediate variables. Precisely, we have a system of $4(\ell+1)n$ quadratic equations (including field equations) in $(\ell+1)n$ variables. For this system of equations, the time complexity of GroverXL is given as $2^{(0.3496+o(1))(\ell+1)n}$ when using $\omega = 2$, which is worse than Grover's algorithm.

The `QuantumBooleanSolve` algorithm [42] is a quantum version of the `BooleanSolve` algorithm [13], which solves a system of Boolean quadratic equations. In [42], its time complexity has been analyzed only for a system of equations with the same number of variables and equations. A single evaluation of AIM can be represented by $4(\ell+1)n$ quadratic equations in $(\ell+1)n$ variables. In this case, the complexity of `QuantumBooleanSolve` is given as $O(2^{0.462(\ell+1)n})$, which is worse than Grover's algorithm.

In contrast to the algorithms discussed above, Chen and Gao [25] proposed a quantum algorithm to solve a system of multivariate equations using the Harrow-Hassidim-Lloyd (HHL) algorithm [51] that solves a sparse system of linear equations with exponential speedup. In brief, Chen and Gao's algorithm solves a system of linear equations from the Macaulay matrix by the HHL algorithm. It has been claimed that this algorithm enjoys exponential speedup for a certain set of parameters. When applied to AIM, the hamming weight of the secret key should be smaller than $O(\log n)$ to achieve exponential speedup [31]. Otherwise, this algorithm is slower than Grover's algorithm [31].

Quantum Generic Attack. A generic attack does not use any particular property of the underlying components (e.g., S-boxes for AIM). The underlying smaller primitives are typically modeled as public random permutations or functions. The Even-Mansour cipher [41], the FX-construction [61] and a Feistel cipher [71] have been analyzed in the classic and generic attack model. As their quantum analogues, the Even-Mansour cipher [64, 21], the FX-construction [65, 52] and a Feistel cipher [63] have been analyzed in the Q1 or Q2 model. Most of these attacks can be seen as a combination of Simon's period finding algorithm [85] (in the Q2 model), and Grover's/offline Simon's algorithms [21] (in the Q1 model). Since Simon's period finding algorithm requires multiple queries to a *keyed* construction (which is not the case for AIM), we believe that the above attacks do not apply to AIM in a straightforward manner.

[10] In the call for proposals by NIST [75], the security level I, III, V are defined as the strength of AES-128, AES-192, AES-256, respectively, against Grover's algorithm.

5.4 Statistical Attacks

An adversary is allowed to evaluate AIM with an arbitrary input pair $(\mathsf{pt}, \mathsf{iv})$ in an offline manner. However, such an evaluation is independent of the actual secret key pt^*, so the adversary is not able to collect a sufficient amount of statistical data which are related to pt^*. Furthermore, the linear layer of AIM is generated independently at random for every user. For this reason, we believe that our construction is secure against any type of statistical attacks including (impossible) differential, boomerang, and integral attacks.

In the multi-target scenario, an adversary has no information on which users have the same secret. Even for multiple users with the same iv, statistical attacks would not be feasible since all the inputs and their differences are unknown to the adversary. That said, to prevent any unexpected variant of differential and linear cryptanalysis, we summarize a lower bound of the weight of differential and correlation trails in this section.

Differential Cryptanalysis. Since AIM is a key-less primitive, we will estimate the security of AIM against differential cryptanalysis by lower bounding the weight of a differential trail (for example, as in [30]).

Given a function $f : \{0,1\}^n \to \{0,1\}^m$, the *weight* of a differential $(\Delta x, \Delta y) \in \{0,1\}^n \times \{0,1\}^m$ is defined by

$$w_d(\Delta x \xrightarrow{f} \Delta y) \stackrel{\text{def}}{=} n - \log_2 |\{x \in \{0,1\}^n : f(x \oplus \Delta x) \oplus f(x) = \Delta y\}| \,.$$

The weight is not defined if there is no x such that $f(x \oplus \Delta x) \oplus f(x) = \Delta y$. Otherwise, we say that Δx and Δy are *compatible*.

A differential trail is the composition of compatible differentials. For AIM, a differential trail from an input to the output (ignoring the feed-forward) can be represented as follows.

$$Q = \Delta_0 \xrightarrow{\mathsf{Mer}[e_1,\ldots,e_\ell]} \Delta_1 \xrightarrow{\mathsf{Lin}} \Delta_2 \xrightarrow{\mathsf{Mer}[e_*]} \Delta_3.$$

Then the weight of the differential trail Q is defined as

$$w_d(Q) \stackrel{\text{def}}{=} \sum_{i=0}^{2} w_d(\Delta_i \to \Delta_{i+1}).$$

The weight of a Mersenne S-box is determined by the number of solutions to $\mathsf{Mer}[e](x \oplus \Delta x) \oplus \mathsf{Mer}[e](x) = \Delta y$, which is a polynomial equation of degree $2^e - 2$. Therefore, there are at most $2^e - 2$ solutions to this equation, which implies for $\Delta x \neq 0$,

$$w_d(\Delta x \xrightarrow{\mathsf{Mer}[e]} \Delta y) \geq n - \log_2(2^e - 2) \geq n - e.$$

Then we have

$$\begin{aligned} w_d(Q) &= \sum_i w_d(\Delta_i \to \Delta_{i+1}) \\ &\geq \max_{1 \leq i \leq \ell} (n - e_i) = n - e_1. \end{aligned}$$

So, for any differential trail Q, $w_d(Q)$ is close to λ with $\lambda = n$. We note that a trail Q such that $w_d(Q) < \lambda$ never incur a collision, and the existence of such trail does not imply the feasibility of differential cryptanalysis since an adversary is not given a large enough number of plaintext-ciphertext pairs to mount the analysis.

Difference Enumeration Attack. Recently, difference enumeration attacks to LowMC have been proposed [77, 66, 69], which require only a couple of chosen plaintext-ciphertext pairs. In such attacks, an adversary enumerates all possible input and output differences and tries to find a collision and recover the unknown key. This type of attacks work for LowMC since it is based on small S-boxes. So one can easily find all possible differentials in LowMC. On the other hand, AIM is based on n-bit S-boxes, making it infeasible to enumerate all possible differences of each S-box.

Linear Cryptanalysis. In contrast to differential cryptanalysis, security against linear cryptanalysis has been rarely evaluated for key-less primitives since its goal is to retrieve the secret key, not finding a collision or a second-preimage. That said, we lower bound the weight of a correlation trail in Appendix D for completeness in a similar way to differential cryptanalysis.

5.5 Provable Security

In this section, we consider the one-wayness of AIM. More precisely, we will prove the *everywhere preimage resistance* [78] of AIM when the underlying S-boxes are modeled as public random permutations and iv is (implicitly) fixed.[11]

For simplicity, we will assume that $\ell = 2$. The security of AIM with $\ell > 2$ is similarly proved. In the public permutation model and in the single-user setting, AIM is defined as

$$\mathsf{AIM}(\mathsf{pt}) = S_3(A_1 \cdot S_1(\mathsf{pt}) \oplus A_2 \cdot S_2(\mathsf{pt}) \oplus b) \oplus \mathsf{pt}$$

for $\mathsf{pt} \in \{0,1\}^n$, where S_1, S_2, S_3 are independent public random permutations, and A_1 and A_2 are fixed $n \times n$ invertible matrices, and b is a fixed $n \times 1$ vector over $\mathbb{F}_2$.

In the preimage resistance experiment, a computationally unbounded adversary $\mathcal{A}$ with oracle access to S_i, $i = 1, 2, 3$, selects and announces a point $\mathsf{ct} \in \{0,1\}^n$ before making queries to the underlying permutations. After making q forward and backward queries in total, $\mathcal{A}$ obtains a *query history*

$$\mathcal{Q} = \{(i_j, x_j, y_j)\}_{j=1}^{q}$$

such that $S_{i_j}(x_j) = y_j$ and $\mathcal{A}$'s j-th query is either $S_{i_j}(x_j) = y_j$ or $S_{i_j}^{-1}(y_j) = x_j$ for $j = 1, \dots q$. We say that $\mathcal{A}$ *succeeds in finding a preimage of* ct if its query history $\mathcal{Q}$ contains three queries $S_1(x_1) = y_1$, $S_2(x_2) = y_2$ and $S_3(x_3) = y_3$ such that $x_1 = x_2$, $x_3 = A_1 \cdot y_1 \oplus A_2 \cdot y_2 \oplus b$ and $\mathsf{ct} = y_3 \oplus \mathsf{pt}$. In this case, we say that $\mathcal{A}$ wins the preimage-finding game, breaking the one-wayness of AIM. Assuming that $\mathcal{A}$ is information-theoretic, we can prove that $\mathcal{A}$'s winning probability, denoted $\mathbf{Adv}^{\mathsf{epre}}_{\mathsf{AIM}}(q)$, is upper bounded as follows.

$$\mathbf{Adv}^{\mathsf{epre}}_{\mathsf{AIM}}(q) \le \frac{2q}{2^n - q}. \tag{6}$$

Proof of (6). Since $\mathcal{A}$ is information-theoretic, we can assume that $\mathcal{A}$ is deterministic. Furthermore, we assume that $\mathcal{A}$ does not make any redundant query. We will also slightly modify $\mathcal{A}$ so that whenever $\mathcal{A}$ makes a (forward or backward) query to S_1 (resp. S_2) obtaining $S_1(x) = y$ (resp. $S_2(x) = y$), $\mathcal{A}$ makes an additional *forward* query to S_2 (resp. S_1) with x *for free*. This additional query will not degrade $\mathcal{A}$'s preimage-finding advantage since $\mathcal{A}$ is free to ignore it.

An evaluation $\mathsf{AIM}(\mathsf{pt}) = \mathsf{ct}$ consists of three S-box queries. Among the three S-box queries, the lastly asked one is called the *preimage-finding query*. We distinguish two cases.

Case 1. The preimage-finding query is made to either S_1 or S_2. Since $\mathcal{A}$ consecutively obtains a pair of queries of the form $S_1(x) = y_1$ and $S_2(x) = y_2$, any preimage-finding query to either S_1 or S_2 should be forward. If it is $S_1(x)$ (without loss of generality), then there should be queries $S_2(x) = y$ for some y and $S_3(z) = x \oplus \mathsf{ct}$ for some z that have already been made by $\mathcal{A}$. In order for $S_1(x)$ to be the preimage-finding query, it should be the case that

$$S_3(A_1 \cdot S_1(x) \oplus A_2 \cdot S_2(x) \oplus B) = x \oplus \mathsf{ct}$$

or equivalently,

$$S_1(x) = A_1^{-1} \cdot (z \oplus b \oplus A_2 \cdot y)$$

which happens with probability at most $\frac{1}{2^n - q}$. Therefore, the probability of this case is upper bounded by $\frac{q}{2^n - q}$.

[11] We do not claim that the algebraic S-boxes of AIM behave like random permutations. The point of the provable security of AIM is that one cannot break the one-wayness of AIM without exploiting any particular properties of the underlying S-boxes.

Case 2. The preimage-finding query is made to S_3. In order to address this case, we use the notion of a *wish list*, which was first introduced in [7]. Namely, whenever $\mathcal{A}$ makes a pair of queries $S_1(x) = y_1$ and $S_2(x) = y_2$, the evaluation

$$S_3 : A_1 \cdot y_1 \oplus A_2 \cdot y_2 \oplus b \mapsto x \oplus \mathsf{ct}$$

is included in the wish list $\mathcal{W}$. In order for an S_3-query to complete an evaluation $\mathsf{AIM}(\mathsf{pt}) = \mathsf{ct}$ for any pt, at least one "wish" in $\mathcal{W}$ should be made come true. Each evaluation in $\mathcal{W}$ is obtained with probability at most $\frac{1}{2^n-q}$, and $|\mathcal{W}| \leq q$. Therefore, the probability of this case is upper bounded by $\frac{q}{2^n-q}$.

Overall, we can conclude that

$$\mathbf{Adv}^{\mathsf{epre}}_{\mathsf{AIM}}(q) \leq \frac{2q}{2^n - q}.$$

One-wayness in the multi-user setting. In the multi-user setting with u users, $\mathcal{A}$ is given u different target images, where the adversarial goal is to invert any of the target images. In this setting, the adversarial preimage finding advantage is upper bounded by

$$\frac{2uq}{2^n - q}. \tag{7}$$

The proof of (7) follows the same line of argument as the single-user security proof. The difference is that the probability that each query to either S_1 or S_2 becomes the preimage-finding one is upper bounded by $\frac{uq}{2^n-q}$ and the size of the wish list (in the second case) is upper bounded by uq.

We note that the above bound does not mean that AIM provides only the birthday-bound security in the multi-user setting. The straightforward birthday-bound attack is mitigated since AIM is based on a distinct linear layer for every user.

6 Performance Evaluation

Environment. The source codes are developed in C++17, using the GNU C++ 8.4.0 (GNU C 7.5.0 for running the algorithms in the third round submission packages for NIST PQC standardization) compiler with the AVX2 instructions on the Ubuntu 18.04 operating system. A large part of our implementation is taken from the BN++ repository,[12] and we modified its symmetric primitive part to accommodate AIM. All the implementations used in the experiments are compiled at the `-O3` optimization level. For the instantiation of the XOF, we use SHAKE in `XKCP` library.[13] We use SHAKE128 for AIMer-I, and SHAKE256 for AIMer-III and AIMer-V. Our experiments are measured in Intel Xeon E5-1650 v3 @ 3.50GHz with 128 GB memory. For a fair comparison, we measure the execution time for each signature scheme on the same CPU using the `taskset` command with Hyper-Threading and Turbo Boost features disabled.

Performance of AIMer. As mentioned in Section 2.3, AIM has been designed to take full advantage of optimization by repeated multipliers to reduce the number of α values. Due to this technique, the overall performance of the signature scheme is improved in terms of both the signature size and the signing time. The performance of AIMer is summarized in Table 5. Parameter sets (i.e., the number of parties N and the number of parallel repetitions τ) for various security levels are chosen in the same way of [58]. We observe that AIMer enjoys the best trade-off between the signature size and the signing/verification time.

In Table 6, AIMer is compared to the state-of-the-art Rainier signature scheme combined with the BN++ proof system (denoted BN++Rain$_r$, where $r \in \{3, 4\}$) with all the optimizations from [58] applied at the 128-bit security level. AIMer-I enjoys 5.14 to 8.21% shorter signature size than BN++Rain$_3$ with similar signing and verification time. Compared to BN++Rain$_4$, AIMer achieves more significant improvement with 13.98 to 21.15% shorter signature size and 5.59 to 14.84% improved signing and verification performance for all the parameter sets.

[12] https://github.com/IAIK/bnpp_helium_signatures/tree/main/bnpp_rain

[13] https://github.com/XKCP/XKCP

Scheme	N	τ	Sign (ms)	Verify (ms)	Size (B)
AIMer-I	16	33	0.82	0.78	5 904
AIMer-I	57	23	1.82	1.77	4 880
AIMer-I	256	17	5.96	5.90	4 176
AIMer-I	1615	13	29.62	29.17	3 840
AIMer-III	16	49	1.57	1.48	13 080
AIMer-III	64	33	3.86	3.62	10 440
AIMer-III	256	25	10.57	10.42	9 144
AIMer-III	1621	19	58.70	58.10	8 352
AIMer-V	16	65	2.87	2.78	25 152
AIMer-V	62	44	6.60	6.54	19 904
AIMer-V	256	33	19.21	19.19	17 088
AIMer-V	1623	25	98.49	98.64	15 392

Table 5: Performance of AIMer for various parameter sets.

Scheme	N	τ	Sign (ms)	Verify (ms)	Size (B)
BN++Rain$_3$ [58]	16	33	0.83	0.77	6 432
BN++Rain$_3$ [58]	57	23	1.83	1.77	5 248
BN++Rain$_3$ [58]	256	17	5.92	5.94	4 448
BN++Rain$_3$ [58]	1615	13	28.95	28.33	4 048
BN++Rain$_4$ [58]	16	33	0.93	0.86	7 488
BN++Rain$_4$ [58]	57	23	2.09	2.01	5 984
BN++Rain$_4$ [58]	256	17	6.45	6.23	4 992
BN++Rain$_4$ [58]	1615	13	32.85	31.86	4 464
AIMer-I	16	33	0.82	0.78	5 904
AIMer-I	57	23	1.82	1.77	4 880
AIMer-I	256	17	5.96	5.90	4 176
AIMer-I	1615	13	29.62	29.17	3 840

Table 6: Performance of AIMer, BN++Rain$_3$, and BN++Rain$_4$ at 128-bit security level.

Comparison. We compare the performance of AIMer to existing post-quantum signature schemes at the 128-bit security level in Table 7. In the first group, we provide the performance of three algorithms selected as the finalists of the NIST competition for PQC standardization - CRYSTALS-Dilithium [72], Falcon [76], and SPHINCS$^+$ [53].

CRYSTALS-Dilithium and Falcon are lattice-based signature schemes with high efficiency in both bandwidth (signature size plus public key size) and signing/verification time. We implemented SPHINCS$^+$ using the SHAKE256 hash function for a fair comparison between symmetric primitives based signature schemes. Compared to any of the small and the fast variants of SPHINCS$^+$, AIMer obviously provides smaller bandwidth and faster signing time at the cost of slightly slower verification.

In the second group, we compare existing ZKP-based signature schemes based on symmetric primitives: Picnic [87], Limbo [80], Banquet,[14] Rainier,[15] and BN++Rain.[16] In particular, Picnic is one of the alternate candidates of the third round of the NIST competition. For Limbo-AES128, we cited the numbers from the paper as its public implementation is not available (to the best of our knowledge). When the number of

[14] https://github.com/dkales/banquet
[15] https://github.com/IAIK/rainier-signatures
[16] https://github.com/IAIK/bnpp_helium_signatures

parties N is set to 16, these schemes require bandwidth of 12,495 to 30,957 bytes, while AIMer requires 5,936 bytes with comparable performance in signing and verification time.

Scheme	$\lvert pk \rvert$ (B)	$\lvert sig \rvert$ (B)	Sign (ms)	Verify (ms)
Dilithium2 [72]	1312	2 420	0.10	0.03
Falcon-512 [76]	897	690	0.27	0.04
SPHINCS$^+$-128s* [53]	32	7 856	315.74	0.35
SPHINCS$^+$-128f* [53]	32	17 088	16.32	0.97
Picnic1-L1-full [87]	32	30 925	1.16	0.91
Picnic3-L1 [87]	32	12 463	5.83	4.24
Banquet [15]	32	19 776	7.09	5.24
Limbo-AES128† [80]	32	21 520	2.70	2.00
Rainier$_3$ [38]	32	8 544	0.97	0.89
BN++Rain$_3$ [58]	32	6 432	0.83	0.77
AIMer-I	32	5 904	0.82	0.78

*: -SHAKE-simple

†: measurements are from this paper.

Table 7: Comparison of AIMer to existing (post-quantum) signature schemes at 128-bit security level. The number of parties N is set to 16 for ZKP-based signature schemes.

References

[1] Albrecht, M., Grassi, L., Rechberger, C., Roy, A., Tiessen, T.: MiMC: Efficient Encryption and Cryptographic Hashing with Minimal Multiplicative Complexity. In: ASIACRYPT 2016. pp. 191–219. Springer (2016)

[2] 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: ASIACRYPT 2019. pp. 371–397. Springer (2019)

[3] Albrecht, M.R., Rechberger, C., Schneider, T., Tiessen, T., Zohner, M.: Ciphers for MPC and FHE. In: EUROCRYPT 2015. pp. 430–454. Springer (2015)

[4] 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) (Sep 2020)

[5] Amy, M., Maslov, D., Mosca, M., Roetteler, M.: A Meet-in-the-Middle Algorithm for Fast Synthesis of Depth-Optimal Quantum Circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **32**(6), 818–830 (2013)

[6] Aoki, K., Ichikawa, T., Kanda, M., Matsui, M., Moriai, S., Nakajima, J., Tokita, T.: Camellia: A 128-Bit Block Cipher Suitable for Multiple Platforms — Design and Analysis. In: SAC 2000. pp. 39–56. Springer (2001)

[7] Armknecht, F., Fleischmann, E., Krause, M., Lee, J., Stam, M., Steinberger, J.: The Preimage Security of Double-Block-Length Compression Functions. In: ASIACRYPT 2011. pp. 233–251. Springer (2011)

[8] Ashur, T., Dhooghe, S.: MARVELlous: a STARK-Friendly Family of Cryptographic Primitives. Cryptology ePrint Archive (2018)

[9] Banik, S., Barooti, K., Caforio, A., Vaudenay, S.: Memory-Efficient Single Data-Complexity Attacks on LowMC Using Partial Sets. Cryptology ePrint Archive (2022)

[10] Banik, S., Barooti, K., Durak, F.B., Vaudenay, S.: Cryptanalysis of LowMC instances using single plaintext/ciphertext pair. IACR Transactions on Symmetric Cryptology **2020**(4), 130–146 (Dec 2020), https://tosc.iacr.org/index.php/ToSC/article/view/8751

[11] Banik, S., Barooti, K., Vaudenay, S., Yan, H.: New Attacks on LowMC Instances with a Single Plaintext/Ciphertext Pair. In: ASIACRYPT 2021. pp. 303–331. Springer (2021)

[12] Bardet, M., Faugere, J.C., Salvy, B.: On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations. In: Proceedings of the International Conference on Polynomial System Solving. pp. 71–74 (2004)

[13] Bardet, M., Faugère, J.C., Salvy, B., Spaenlehauer, P.J.: On the complexity of solving quadratic Boolean systems. Journal of Complexity **29**(1), 53–75 (2013), https://www.sciencedirect.com/science/article/pii/S0885064X12000611
[14] Baum, C., Nof, A.: Concretely-Efficient Zero-Knowledge Arguments for Arithmetic Circuits and Their Application to Lattice-Based Cryptography. In: PKC 2020. pp. 495–526. Springer (2020)
[15] Baum, C., de Saint Guilhem, C.D., Kales, D., Orsini, E., Scholl, P., Zaverucha, G.: Banquet: Short and Fast Signatures from AES. In: PKC 2021. pp. 266–297. Springer (2021)
[16] Beigel, R.: The polynomial method in circuit complexity. In: [1993] Proceedings of the Eigth Annual Structure in Complexity Theory Conference. pp. 82–95 (1993)
[17] Bernstein, D.J., Yang, B.Y.: Asymptotically Faster Quantum Algorithms to Solve Multivariate Quadratic Equations. In: PQCrypto 2018. pp. 487–506. Springer (2018)
[18] Beullens, W.: Sigma Protocols for MQ, PKP and SIS, and Fishy Signature Schemes. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020. pp. 183–211. Springer International Publishing, Cham (2020)
[19] Beullens, W.: Breaking Rainbow Takes a Weekend on a Laptop. In: Dodis, Y., Shrimpton, T. (eds.) Advances in Cryptology – CRYPTO 2022. pp. 464–479. Springer Nature Switzerland, Cham (2022)
[20] Beullens, W., Delpech de Saint Guilhem, C.: LegRoast: Efficient Post-quantum Signatures from the Legendre PRF. In: International Conference on Post-Quantum Cryptography. pp. 130–150. Springer (2020)
[21] Bonnetain, X., Hosoyamada, A., Naya-Plasencia, M., Sasaki, Y., Schrottenloher, A.: Quantum Attacks Without Superposition Queries: The Offline Simon's Algorithm. In: ASIACRYPT 2019. pp. 552–583. Springer (2019)
[22] Bosma, W., Cannon, J., Playoust, C.: The Magma Algebra System I: The User Language. J. Symbolic Comput. **24**(3-4), 235–265 (1997), http://dx.doi.org/10.1006/jsco.1996.0125, Computational algebra and number theory (London, 1993)
[23] Castryck, W., Decru, T.: An efficient key recovery attack on SIDH. In EUROCRYPT 2023, to appear (2023)
[24] Chase, M., Derler, D., Goldfeder, S., Orlandi, C., Ramacher, S., Rechberger, C., Slamanig, D., Zaverucha, G.: Post-Quantum Zero-Knowledge and Signatures from Symmetric-Key Primitives. In: ACM CCS 2017. pp. 1825–1842 (2017)
[25] Chen, Y.A., Gao, X.S.: Quantum Algorithm for Boolean Equation Solving and Quantum Algebraic Attack on Cryptosystems. Journal of Systems Science and Complexity **35**(1), 373–412 (Feb 2022), https://doi.org/10.1007/s11424-020-0028-6
[26] Cheon, J.H., Lee, D.H.: Resistance of S-Boxes against Algebraic Attacks. In: FSE 2004. pp. 83–93. Springer (2004)
[27] Courtois, N., Klimov, A., Patarin, J., Shamir, A.: Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In: EUROCRYPT 2000. pp. 392–407. Springer (2000)
[28] Courtois, N.T., Debraize, B., Garrido, E.: On Exact Algebraic [Non-]Immunity of S-Boxes Based on Power Functions. In: ACISP 2006. pp. 76–86. Springer (2006)
[29] Daemen, J., Rijmen, V.: The Design of Rijndael, vol. 2. Springer (2002)
[30] Daemen, J., Van Assche, G.: Differential Propagation Analysis of Keccak. In: Canteaut, A. (ed.) Fast Software Encryption. pp. 422–441. Springer (2012)
[31] Ding, J., Gheorghiu, V., Gilyén, A., Hallgren, S., Li, J.: Limitations of the Macaulay matrix approach for using the HHL algorithm to solve multivariate polynomial systems. arXiv 2111.00405 (2021), https://arxiv.org/abs/2111.00405
[32] Ding, J., Schmidt, D.: Rainbow, a new multivariable polynomial signature scheme. In: ACNS 2005. pp. 164–175. Springer (2005)
[33] Ding, J., Schmidt, D.: Solving Degree and Degree of Regularity for Polynomial Systems over a Finite Fields, pp. 34–49. Springer (2013)
[34] Dinur, I.: Cryptanalytic Applications of the Polynomial Method for Solving Multivariate Equation Systems over GF(2). In: Canteaut, A., Standaert, F.X. (eds.) Advances in Cryptology – EUROCRYPT 2021. pp. 374–403. Springer, Cham (2021)
[35] Dinur, I., Goldfeder, S., Halevi, T., Ishai, Y., Kelkar, M., Sharma, V., Zaverucha, G.: MPC-Friendly Symmetric Cryptography from Alternating Moduli: Candidates, Protocols, and Applications. In: CRYPTO 2021. pp. 517–547. Springer (2021)
[36] Dinur, I., Liu, Y., Meier, W., Wang, Q.: Optimized Interpolation Attacks on LowMC. In: ASIACRYPT 2015. vol. 9453, pp. 535–560. Springer (2015)
[37] Dobraunig, C., Eichlseder, M., Mendel, F.: Higher-Order Cryptanalysis of LowMC. In: ICISC 2015. vol. 9558, pp. 87–101. Springer (2016)

[38] 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. p. 843–857. CCS '22, Association for Computing Machinery, New York, NY, USA (2022), https://doi.org/10.1145/3548606.3559353

[39] Don, J., Fehr, S., Majenz, C.: The Measure-and-Reprogram Technique 2.0: Multi-Round Fiat-Shamir and More. In: CRYPTO 2020. p. 602–631. Springer (2020)

[40] 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: ASIACRYPT 2020. pp. 477–506. Springer (2020)

[41] Even, S., Mansour, Y.: A construction of a cipher from a single pseudorandom permutation. Journal of Cryptology **10**(3), 151–161 (Jun 1997), https://doi.org/10.1007/s001459900025

[42] Faugère, J.C., Horan, K., Kahrobaei, D., Kaplan, M., Kashefi, E., Perret, L.: Fast Quantum Algorithm for Solving Multivariate Quadratic Equations. Cryptology ePrint Archive, Paper 2017/1236 (2017), https://eprint.iacr.org/2017/1236

[43] Feneuil, T., Joux, A., Rivain, M.: Syndrome Decoding in the Head: Shorter Signatures from Zero-Knowledge Proofs. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022. pp. 541–572. Springer Nature Switzerland, Cham (2022)

[44] Feneuil, T., Maire, J., Rivain, M., Vergnaud, D.: Zero-Knowledge Protocols for the Subset Sum Problem from MPC-in-the-Head with Rejection. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology – ASIACRYPT 2022. pp. 371–402. Springer Nature Switzerland, Cham (2022)

[45] Fröberg, R.: An Inequality for Hilbert Series of Graded Algebras. MATHEMATICA SCANDINAVICA **56** (Dec 1985)

[46] Giacomelli, I., Madsen, J., Orlandi, C.: ZKBoo: Faster Zero-Knowledge for Boolean Circuits. In: USENIX Security 2016. pp. 1069–1083. USENIX Association (2016)

[47] Grassi, L., Khovratovich, D., Rechberger, C., Roy, A., Schofnegger, M.: Poseidon: A New Hash Function for Zero-Knowledge Proof Systems. In: USENIX Security 2021. pp. 519–535. USENIX Association (2021)

[48] Grassi, L., Lüftenegger, R., Rechberger, C., Rotaru, D., Schofnegger, M.: On a Generalization of Substitution-Permutation Networks: The HADES Design Strategy. In: EUROCRYPT 2020. pp. 674–704. Springer (2020)

[49] Grover, L.K.: A Fast Quantum Mechanical Algorithm for Database Search. In: ACM STOC '96. p. 212–219. Association for Computing Machinery (1996)

[50] Gupta, K.C., Ray, I.G.: Finding Biaffine and Quadratic Equations for S-Boxes Based on Power Mappings. IEEE Transactions on Information Theory **61**(4), 2200–2209 (2015)

[51] Harrow, A.W., Hassidim, A., Lloyd, S.: Quantum Algorithm for Linear Systems of Equations. Phys. Rev. Lett. **103**, 150502 (Oct 2009)

[52] Hosoyamada, A., Sasaki, Y.: Cryptanalysis Against Symmetric-Key Schemes with Online Classical Queries and Offline Quantum Computations. In: CT-RSA 2018. pp. 198–218. Springer (2018)

[53] Hulsing, A., Bernstein, D.J., Dobraunig, C., Eichlseder, M., Fluhrer, S., Gazdag, S.L., Kampanakis, P., Kolbl, S., Lange, T., Lauridsen, M.M., Mendel, F., Niederhagen, R., Rechberger, C., Rijneveld, J., Schwabe, P., Aumasson, J.P., Westerbaan, B., Beullens, W.: SPHINCS+. Technical report, National Institute of Standards and Technology, 2022 (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022

[54] Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge from Secure Multiparty Computation. In: ACM STOC 2007. pp. 21–30 (2007)

[55] Jang, K., Baksi, A., Kim, H., Seo, H., Chattopadhyay, A.: Improved Quantum Analysis of SPECK and LowMC. In: Isobe, T., Sarkar, S. (eds.) Progress in Cryptology – INDOCRYPT 2022. pp. 517–540. Springer International Publishing, Cham (2022)

[56] Jang, K., Kim, W., Lim, S., Kang, Y., Yang, Y., Seo, H.: Optimized Implementation of Quantum Binary Field Multiplication with Toffoli Depth One. In: WISA 2022: Information Security Applications. Springer (2022)

[57] Jao, D., Azarderakhsh, R., Campagna, M., Costello, C., De Feo, L., Hess, B., Jalili, A., Koziel, B., LaMacchia, B., Longa, P., Naehrig, M., Renes, J., Soukharev, V., Urbanik, D.: SIKE: Supersingular Isogeny Key Encapsulation. HAL **2017**(0) (2017), http://dml.mathdoc.fr/item/hal-02171951

[58] Kales, D., Zaverucha, G.: Efficient Lifting for Shorter Zero-Knowledge Proofs and Post-Quantum Signatures. Cryptology ePrint Archive, Paper 2022/588 (2022), https://eprint.iacr.org/2022/588

[59] Katz, D.J., Schmidt, K., Winterhof, A.: Weil sums of binomials: Properties applications and open problems. In: Combinatorics and Finite Fields: Difference Sets, Polynomials, Pseudorandomness and Applications, vol. 23, pp. 109–134. De Gruyter (2019)

[60] Katz, J., Kolesnikov, V., Wang, X.: Improved Non-Interactive Zero Knowledge with Applications to Post-Quantum Signatures. In: ACM CCS 2018. pp. 525–537. ACM (2018)

[61] Kilian, J., Rogaway, P.: How to Protect DES Against Exhaustive Key Search (an Analysis of DESX). Journal of Cryptology **14**(1), 17–35 (Jan 2001), https://doi.org/10.1007/s001450010015

[62] Kipnis, A., Shamir, A.: Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization. In: CRYPTO '99. pp. 19–30. Springer (1999)

[63] Kuwakado, H., Morii, M.: Quantum distinguisher between the 3-round Feistel cipher and the random permutation. In: 2010 IEEE International Symposium on Information Theory. pp. 2682–2685 (2010)

[64] Kuwakado, H., Morii, M.: Security on the quantum-type Even-Mansour cipher. In: 2012 International Symposium on Information Theory and its Applications. pp. 312–316 (2012)

[65] Leander, G., May, A.: Grover Meets Simon – Quantumly Attacking the FX-construction. In: ASIACRYPT 2017. pp. 161–178. Springer (2017)

[66] Liu, F., Isobe, T., Meier, W.: Cryptanalysis of full LowMC and LowMC-M with Algebraic Techniques. In: CRYPTO 2021. pp. 368–401. Springer (2021)

[67] Liu, F., Isobe, T., Meier, W.: Low-Memory Algebraic Attacks on Round-Reduced LowMC. Cryptology ePrint Archive (2021)

[68] Liu, F., Meier, W., Sarkar, S., Isobe, T.: New Low-Memory Algebraic Attacks on LowMC in the Picnic Setting. IACR Transactions on Symmetric Cryptology **2022**(3), 102–122 (Sep 2022), https://tosc.iacr.org/index.php/ToSC/article/view/9851

[69] Liu, F., Sarkar, S., Wang, G., Meier, W., Isobe, T.: Algebraic Meet-in-the-Middle Attack on LowMC. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology – ASIACRYPT 2022. pp. 225–255. Springer Nature Switzerland, Cham (2022)

[70] Lokshtanov, D., Paturi, R., Tamaki, S., Williams, R., Yu, H.: Beating Brute Force for Systems of Polynomial Equations over Finite Fields. In: Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 2190–2202. SIAM (2017)

[71] Luby, M., Rackoff, C.: How to Construct Pseudo-random Permutations from Pseudo-random Functions. In: CRYPTO '85. pp. 447–447. Springer (1986)

[72] Lyubashevsky, V., Ducas, L., Kiltz, E., Lepoint, T., Schwabe, P., Seiler, G., Stehlé, D., Bai, S.: CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2022 (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022

[73] Muñoz-Coreas, E., Thapliyal, H.: Design of Quantum Circuits for Galois Field Squaring and Exponentiation. In: 2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). pp. 68–73 (2017)

[74] Nawaz, Y., Gupta, K.C., Gong, G.: Algebraic Immunity of S-Boxes Based on Power Mappings: Analysis and Construction. IEEE Transactions on Information Theory **55**(9), 4263–4273 (2009)

[75] NIST: Call for Additional Digital Signature Schemes for the Post-Quantum Cryptography Standardization Process. Technical report, National Institute of Standards and Technology, 2022 (2022), available at https://csrc.nist.gov/projects/pqc-dig-sig

[76] Prest, T., Fouque, P.A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Ricosset, T., Seiler, G., Whyte, W., Zhang, Z.: FALCON. Technical report, National Institute of Standards and Technology, 2022 (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022

[77] Rechberger, C., Soleimany, H., Tiessen, T.: Cryptanalysis of Low-Data Instances of Full LowMCv2. IACR Transactions on Symmetric Cryptology **2018**(3), 163–181 (2018)

[78] Rogaway, P., Shrimpton, T.: Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In: FSE 2004. pp. 371–388. Springer (2004)

[79] de Saint Guilhem, C.D., Meyer, L.D., Orsini, E., Smart, N.P.: BBQ: Using AES in Picnic Signatures. In: SAC 2019. pp. 669–692. Springer (2019)

[80] de Saint Guilhem, C.D., Orsini, E., Tanguy, T.: Limbo: Efficient Zero-Knowledge MPCitH-Based Arguments. In: ACM CCS 2021. p. 3022–3036. Association for Computing Machinery (2021)

[81] Sauer, J.F., Szepieniec, A.: SoK: Gröbner Basis Algorithms for Arithmetization Oriented Ciphers. Cryptology ePrint Archive, Paper 2021/870 (2021), https://eprint.iacr.org/2021/870

[82] Schwabe, P., Avanzi, R., Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Seiler, G., Stehlé, D., Ding, J.: CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2022 (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022

[83] Shirai, T., Shibutani, K., Akishita, T., Moriai, S., Iwata, T.: The 128-Bit Blockcipher CLEFIA (Extended Abstract). In: FSE 2007. pp. 181–195. Springer (2007)

[84] Shor, P.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th Annual Symposium on Foundations of Computer Science. pp. 124–134 (1994)
[85] Simon, D.R.: On the Power of Quantum Computation. SIAM Journal on Computing **26**(5), 1474–1483 (1997), https://doi.org/10.1137/S0097539796298637
[86] Williams, R.R.: The Polynomial Method in Circuit Complexity Applied to Algorithm Design (Invited Talk). In: Raman, V., Suresh, S.P. (eds.) 34th International Conference on Foundation of Software Technology and Theoretical Computer Science (FSTTCS 2014). Leibniz International Proceedings in Informatics (LIPIcs), vol. 29, pp. 47–60. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2014), http://drops.dagstuhl.de/opus/volltexte/2014/4832
[87] Zaverucha, G., Chase, M., Derler, D., Goldfeder, S., Orlandi, C., Ramacher, S., Rechberger, C., Slamanig, D., Katz, J., Wang, X., Kolesnikov, V., Kales, D.: Picnic. Technical report, National Institute of Standards and Technology, 2020 (2022), available at https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-3-submissions
[88] Zhandry, M.: How to Construct Quantum Random Functions. In: 2012 IEEE 53rd Annual Symposium on Foundations of Computer Science. pp. 679–687 (2012)

A Actual Running time of Gröbner Basis Computation

Figure 4 shows the actual running time of Gröbner basis computation for single-round Even-Mansour ciphers built on top of three S-boxes: NGG, Mersenne, and the inverse. We observe that Gröbner basis computation becomes faster given a larger number of quadratic equations. For a fixed S-box, the computation is faster with the full system than the basic system.

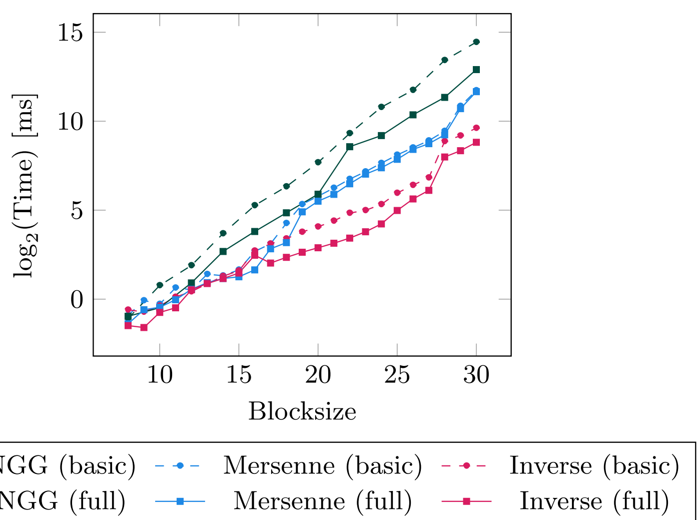

Fig. 4: Gröbner basis computation time for single-round Even-Mansour ciphers based on NGG, Mersenne and the inverse S-boxes. This experiment is done in AMD Ryzen 7 2700X @ 3.70GHz with 128 GB memory.

B Systems of Equations from AIM

There are multiple ways of building a system of equations from an evaluation of AIM. We can categorize them according to the number of (Boolean) variables and find the optimal choice of variables to obtain a system of the lowest degree. Since $\ell \in \{2, 3\}$ is recommended, we consider 4 types of systems of equations as follows.

1. Systems in n variables.
2. Systems in $2n$ variables.
3. Systems in $3n$ variables.
4. Systems in $4n$ variables (only for AIM-V).

Using the quadratic relation between an input and the output of each Mersenne S-box, we can establish a system of *quadratic* equations in $(\ell+1)n$ variables. With fewer variables, the resulting systems would have higher degrees. The goal of this section is to find a system of equations of the lowest degree for each type, where such systems of equations are denoted $S_1, S_2, \ldots, S_{\mathsf{quad}}$, respectively. The optimal systems of equations will be defined using the following variables.

- x: the input of AIM, i.e., pt
- y_i: the output of $\mathsf{Mer}[e_i]$ for $i = 1, \ldots, \ell$
- z: the output of Lin

The underlying $\ell+1$ Mersenne S-boxes determine explicit and implicit relations between these variables. For example, $\mathsf{Mer}[e_i]$ implicitly determines $3n$ quadratic equations in x and y_i, while y_i (resp. x) can be explicitly represented by a polynomial in x (resp. y_i). We can also explicitly represent y_i using y_j for $j \neq i$ or z as follows.

$$\begin{aligned} y_i &= \mathsf{Mer}[e_i] \circ \mathsf{Mer}[e_j]^{-1}(y_j), \\ &= \mathsf{Mer}[e_i]\left(\mathsf{Mer}[e_*](z) \oplus \mathsf{ct}\right). \end{aligned}$$

The degree of y_i with respect to z might be greater than the degree of $\mathsf{Mer}[e_i] \circ \mathsf{Mer}[e_*]$ due to the constant addition, while we will ignore the effect by ct for simplicity. Table 8 shows the degrees of all the possible explicit relations from AIM, and this table can be used to find the optimal systems of equations.

Relation	AIM-I	AIM-III	AIM-V
$\mathsf{Mer}[e_1]$	3	5	3
$\mathsf{Mer}[e_1]^{-1}$	43	77	171
$\mathsf{Mer}[e_2]$	27	29	53
$\mathsf{Mer}[e_2]^{-1}$	19	53	29
$\mathsf{Mer}[e_2] \circ \mathsf{Mer}[e_1]^{-1}$	9	121	103
$\mathsf{Mer}[e_1] \circ \mathsf{Mer}[e_2]^{-1}$	57	73	87
$\mathsf{Mer}[e_*]$	5	7	5
$\mathsf{Mer}[e_*]^{-1}$	77	55	205
$\mathsf{Mer}[e_1] \circ \mathsf{Mer}[e_*]$	5	7	5
$\mathsf{Mer}[e_2] \circ \mathsf{Mer}[e_*]$	27	29	53
$\mathsf{Mer}[e_*]^{-1} \circ \mathsf{Mer}[e_1]^{-1}$	67	78	171
$\mathsf{Mer}[e_*]^{-1} \circ \mathsf{Mer}[e_2]^{-1}$	64	100	110
$\mathsf{Mer}[e_3]$	-	-	7
$\mathsf{Mer}[e_3]^{-1}$	-	-	183
$\mathsf{Mer}[e_3] \circ \mathsf{Mer}[e_1]^{-1}$	-	-	173
$\mathsf{Mer}[e_3] \circ \mathsf{Mer}[e_2]^{-1}$	-	-	203
$\mathsf{Mer}[e_1] \circ \mathsf{Mer}[e_3]^{-1}$	-	-	37
$\mathsf{Mer}[e_2] \circ \mathsf{Mer}[e_3]^{-1}$	-	-	227
$\mathsf{Mer}[e_3] \circ \mathsf{Mer}[e_*]$	-	-	7
$\mathsf{Mer}[e_*]^{-1} \circ \mathsf{Mer}[e_3]^{-1}$	-	-	125

Table 8: Degrees of the compositions and the inverses of the Mersenne S-boxes of AIM.

Scheme	Name	#Var	Variables	(#Eq, Deg)	Gröbner Basis		XL		Dinur [34]	
					d_{reg}	Time	D	Time	Time	Memory
AIM-I	S_1	n	z	$(3n, 10)$	51	300.8	52	244.8	**137.3**	138.3
	S_2	$2n$	$x,\ y_2$	$(3n, 2) + (3n, 4)$	22	**214.9**	14	150.4	248.3	253.7
	S_{quad}	$3n$	$x,\ y_1,\ y_2$	$(9n, 2)$	20	222.8	12	**148.0**	330.1	346.3
AIM-III	S_1	n	z	$(3n, 14)$	82	474.0	84	375.3	**202.1**	203.3
	S_2	$2n$	$x,\ y_2$	$(3n, 2) + (3n, 6)$	31	**310.6**	18	203.0	377.5	382.9
	S_{quad}	$3n$	$x,\ y_1,\ y_2$	$(9n, 2)$	27	310.8	15	**194.1**	487.7	512.1
AIM-V	S_1	n	z	$(3n, 12)$	100	601.1	101	489.7	**264.1**	265.9
	S_2	$2n$	$x,\ y_2$	$(3n, 2) + (3n, 8)$	40	**406.2**	26	289.5	506.3	511.7
	S_3	$3n$	$x,\ y_2,\ y_3$	$(6n, 2) + (3n, 4)$	47	510.4	20	**260.6**	716.1	732.3
	S_{quad}	$4n$	$x,\ y_1,\ y_2,\ y_3$	$(12n, 2)$	45	530.3	19	266.1	854.4	897.7

Table 9: Optimal systems of equations and their security against algebraic attacks. $(\#\mathrm{Eq}, \mathrm{Deg}) = (a, b)$ means that the system contains a equations of degree b. The degree of regularity (resp. the target degree) of the system is denoted d_{reg} (resp. D). The time and the memory complexities are estimated in bits.

After exhaustive search, we found the optimal systems S_1, S_2, S_3 and S_{quad}. First, in order to obtain the S_1 systems, choose z as an n-bit variable. Then x and y_i can be represented as polynomials in z; x is of degree e_*, y_1 is of degree $\deg(\mathsf{Mer}[e_1] \circ \mathsf{Mer}[e_*])$, and y_3 (only for AIM-V) is of degree $\deg(\mathsf{Mer}[e_3] \circ \mathsf{Mer}[e_*])$ with respect to z. Let Lin' denote a linear function such that $y_2 = \mathsf{Lin}'(y_1, y_3, z)$ (which is uniquely determined by Lin). Then we have the following equation.

$$\begin{aligned}(\mathsf{Mer}[e_*](z) \oplus \mathsf{ct})^{2^{e_2}} &= (\mathsf{Mer}[e_*](z) \oplus \mathsf{ct}) \\ &\times \mathsf{Lin}'(\mathsf{Mer}[e_1](\mathsf{Mer}[e_*](z) \oplus \mathsf{ct}), \mathsf{Mer}[e_3](\mathsf{Mer}[e_*](z) \oplus \mathsf{ct}), z).\end{aligned}$$

Since every Mersenne S-box used in AIM is represented by $3n$ quadratic equations, the above system of equations can be seen as a system of $3n$ (Boolean) equations of degree

$$e_* + \max\left(\deg(\mathsf{Mer}[e_1] \circ \mathsf{Mer}[e_*]), \deg(\mathsf{Mer}[e_3] \circ \mathsf{Mer}[e_*])\right).$$

Second, in order to obtain the S_2 systems, we begin with x and y_2, and using $y_1 = \mathsf{Mer}[e_1](x)$ and $y_3 = \mathsf{Mer}[e_3](x)$ (only for AIM-V), we establish the following system of equations.

$$\begin{aligned}&x \cdot y_2 = x^{2^{e_2}}, \\ &\mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, \mathsf{Mer}[e_3](x)) \cdot (x \oplus \mathsf{ct}) \\ &= \mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, \mathsf{Mer}[e_3](x))^{2^{e_*}}\end{aligned}$$

We note that $3n$ quadratic equations are obtained from the first equation, and $3n$ equations of degree $\max(e_1, e_3) + 1$ from the second one.

Third, in order to obtain the S_3 system for AIM-V, we begin with x, y_2 and y_3, and using $y_1 = \mathsf{Mer}[e_1](x)$, we establish the following system of equations.

$$\begin{aligned}x \cdot y_2 &= x^{2^{e_2}} \\ x \cdot y_3 &= x^{2^{e_3}} \\ \mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, y_3) \cdot (x \oplus \mathsf{ct}) &= \mathsf{Lin}(\mathsf{Mer}[e_1](x), y_2, y_3)^{2^{e_*}}.\end{aligned}$$

We note that $6n$ quadratic equations are obtained from the first and the second equations, and $3n$ equations of degree $e_1 + 1$ are from the third one.

Finally, the S_{quad} systems are quadratic with x and all the y_i's being variables. Using the implicit relations for all $\ell+1$ S-boxes, we establish the following system of equations.

$$\begin{aligned}
x \cdot y_1 &= x^{2^{e_1}} \\
x \cdot y_2 &= x^{2^{e_2}} \\
&\vdots \\
x \cdot y_\ell &= x^{2^{e_\ell}} \\
\mathsf{Lin}(y_1, y_2, \ldots, y_\ell) \cdot (x \oplus \mathsf{ct}) &= \mathsf{Lin}(y_1, y_2, \ldots, y_\ell)^{2^{e_*}},
\end{aligned}$$

which can be extended to a system of $3(\ell+1)n$ quadratic equations in $(\ell+1)n$ variables.

Table 9 summarizes the number of variables, the number of equations, and their degrees for the optimal systems of equations, and their security against the Gröbner basis attack, the XL attack, and Dinur's algorithm based on the polynomial method [34]. The XL attack has been discussed in Section 2.2, while more careful analysis is needed when the system of equations consists of equations of different degrees with a particular structure. For example, the S_2 system of AIM-V consists of two types of equations of different degrees: $3n$ equations of degree 2, and $3n$ equations of degree d, all in x and y_2, where $d = \max(e_1, e_3) + 1$. We observe that each type of equations have 2^n solutions since y_2 is uniquely determined for each x, and this property makes one to compute the target degree in a different way from the one given in Section 2.2.

With target degree D, the extended system of equations for S_2 is represented as

$$M\mathbf{v} = \begin{bmatrix} M_2 \\ \hdashline M_* \end{bmatrix} \mathbf{v} = \mathbf{c}$$

where $\mathbf{v}$ is a vector of monomials of degree at most D in x and y_2, M_2 (resp. M_*) is the matrix whose rows are the coefficients of the extended system from $\mathsf{Mer}[e_2]$ (resp. $\mathsf{Mer}[e_*]$), and $\mathbf{c}$ is the corresponding constant vector. The number of rows of M_2 is greater than that of M_* since the original system from $\mathsf{Mer}[e_2]$ has a lower degree than $\mathsf{Mer}[e_*]$. In order for the XL attack to work with the target degree D, the matrix M should have full rank and the number of rows should not be smaller than the number of columns, so that $\mathbf{v}$ is uniquely determined.

On the other hand, the submatrix M_2 itself cannot have full rank since $M_2\mathbf{v} = \mathbf{c}$ should have 2^n solutions (one for each x) as its original system from $\mathsf{Mer}[e_2]$ does. More precisely, the nullity of M_2 should not be smaller than $\sum_{j=1}^{D} \binom{n}{j}$. Otherwise, it implies that there is a linear relation on the monomials consisting of only x variables, for example,

$$\sum_{\mathbf{a}} c_{\mathbf{a}} \mathbf{x}^{\mathbf{a}} = 0$$

where $\mathbf{x}^{\mathbf{a}} = \prod_{i=1}^{n} x_i^{a_i}$ for $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{a} = (a_1, \ldots, a_n)$ such that $\sum_{i=1}^{n} a_i \le D$, and $c_{\mathbf{a}}$ is a Boolean constant. This relation cannot hold for all $\mathbf{x}$, which is a contradiction. Then, for M to have full rank, the rank of M_* should be at least the nullity of M_2, yielding a necessary condition that the number of rows of M_* should be at least $\sum_{j=1}^{D} \binom{n}{j}$ provided that M has no nonzero column.[17] As the number of rows of M_* is the number of equations in the extended system from $\mathsf{Mer}[e_*]$, the target degree D should satisfy the following.

$$3n \cdot \sum_{j=0}^{D-d} \binom{2n}{j} \ge \sum_{j=1}^{D} \binom{n}{j}. \tag{8}$$

For the S_2 system of AIM-III and AIM-V, the target degree is determined by the minimum D satisfying (8), whereas it is not for AIM-I. The difference comes from the small value of $d = 4$ of AIM-I compared to

[17] This condition is satisfied by the assumption that all monomials of degrees up to D appear in the extended system, which can be assumed in the case of AIM.

$d = 6$ and $d = 8$ of AIM-III and AIM-V, respectively. A similar argument also holds for the S_3 system of AIM-V, but it does not determine the target degree either due to the small value of $d = 4$ in S_3.

C Gröbner Basis Computation for AIM and Rain with Toy Parameters

To justify our claim of the theoretical impact of ν, we computed Gröbner bases for the quadratic systems of AIM with $\ell = 2, 3$, and Rain$_3$ with toy parameters.[18] We summarize the result in Figure 5. Similarly to the case of an Even-Mansour cipher, the solving degrees for both basic and full systems are close to the estimated values for the full system. This result suggests that the exact number of quadratic equations should be estimated by the degree of regularity for the full system.

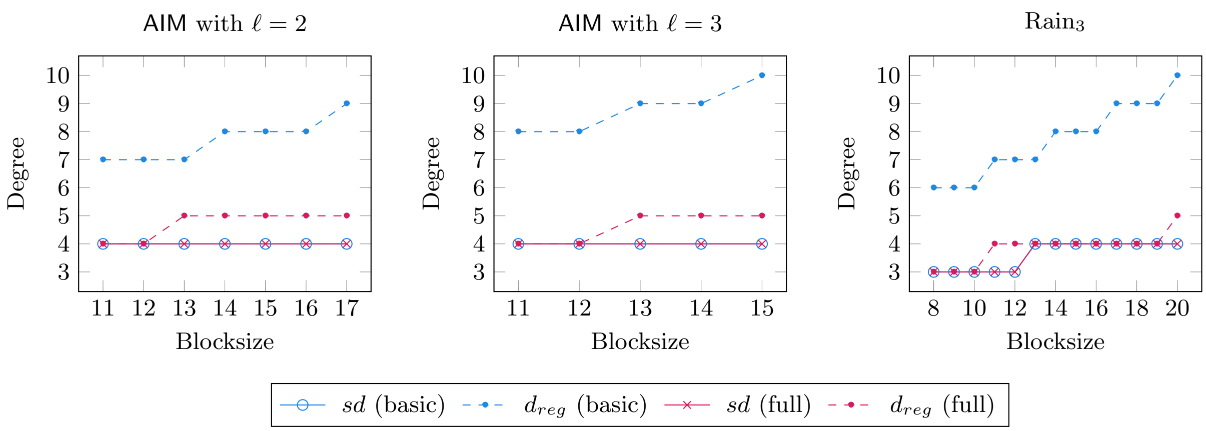

Fig. 5: The estimated degree of regularity d_{reg} and the solving degree sd for the quadratic systems of AIM with $\ell \in \{2, 3\}$ and Rain$_3$.

D Linear Cryptanalysis on AIM

Given a function $f : \{0,1\}^n \to \{0,1\}^m$, the *weight* of a correlation $(\alpha, \beta) \in \{0,1\}^n \times \{0,1\}^m$ is defined by

$$w_l(\alpha \xrightarrow{f} \beta) \stackrel{\text{def}}{=} n - \log_2 \left|2\left|\{x \in \{0,1\}^n : \alpha^\top x = \beta^\top f(x)\}\right| - 2^n\right|.$$

The weight is not defined if there are exactly 2^{n-1} values for x such that $\alpha^\top x = \beta^\top f(x)$. Otherwise, we say that α and β are *compatible*.

A correlation trail is the composition of compatible correlations. For AIM, a correlation trail from an input to the output (ignoring the feed-forward) can be represented as follows.

$$Q = \alpha_0 \xrightarrow{\mathsf{Mer}[e_1,\ldots,e_\ell]} \alpha_1 \xrightarrow{\mathsf{Lin}} \alpha_2 \xrightarrow{\mathsf{Mer}[e_*]} \alpha_3.$$

Then the weight of the correlation trail Q is defined as

$$w_l(Q) \stackrel{\text{def}}{=} \sum_{i=0}^{2} w_l(\alpha_i \to \alpha_{i+1}).$$

[18] We also tried to perform the XL attack on AIM with toy parameters, but could not obtain meaningful data due to the memory limit.

When d is not a power-of-2 and $f(x) = x^d$ is invertible over $\mathbb{F}_{2^n}$, one has the following generic bound [59].

$$\left|2\left|\{x : \alpha^\top x = \beta^\top f(x)\}\right| - 2^n\right| \leq (d-1)2^{n/2}$$

for any compatible correlation (α, β). Therefore the weight of a correlation trail of a Mersenne S-box is lower bounded by $w_l(Q) \geq \frac{n}{2} - e$. Then we have

$$\begin{aligned} w_l(Q) &= \sum_i w_l(\alpha_i \to \alpha_{i+1}) \\ &\geq \max_{1\leq i\leq \ell}(n/2 - e_i) + w_l(\alpha_2 \to \alpha_3) \\ &\geq \max_{1\leq i\leq \ell}(n/2 - e_i) + (n/2 - e_*) \\ &= n - e_1 - e_*. \end{aligned}$$

As Lin is a (full-rank) compression function, α_2 cannot be the zero mask. Since linear cryptanalysis requires $2^{2w_l(Q)}$ plaintext-ciphertext pairs, AIM would be secure against linear cryptanalysis if

$$2(n - e_1 - e_*) \geq \lambda$$

which is the case for AIM. We emphasize again that linear cryptanalysis is not practically relevant in our setting since AIM does not use any secret key, while all the inputs are kept secret and every user is assigned a distinct linear layer.

E Full Description of AIMer

The AIMer signature scheme consists of three algorithms: key generation, signing, and verification algorithms. The key generation takes as input a security parameter and outputs a public key $(\mathsf{iv}, \mathsf{ct})$ and a secret key pt such that $\mathsf{ct} = \mathsf{AIM}(\mathsf{iv}, \mathsf{pt})$. The signing algorithm takes as input the pair of secret and public keys $(\mathsf{pt}, (\mathsf{iv}, \mathsf{ct}))$ and a message m and outputs the corresponding signature σ. The verification algorithm takes as input the public key $(\mathsf{iv}, \mathsf{ct})$, a message m and a signature σ and outputs either Accept or Reject. We describe the AIMer signing and verification algorithms in Algorithm 1 and 2, respectively.

The BN++ proof system is combined with AIM, yielding the AIMer signature scheme. The AIM function has been designed to fully exploit the optimization techniques of the BN++ proof system using *repeated multipliers* for checking multiplication triples and *locally computed output shares* to reduce the overall signature.

Repeated Multiplier. If multiplication triples share the same multiplier, then the α values in the multiplication checking protocol can be batched as mentioned in Section 2.3. The $\ell + 1$ S-box evaluations in AIM produce the $\ell + 1$ multiplication triples that needs to be verified, reformulated as follows.

$$\mathsf{pt} \cdot t_i = \mathsf{pt}^{2^{e_i}}$$

for $i = 1, \ldots, \ell$, and

$$\mathsf{pt} \cdot \mathsf{Lin}[\mathsf{iv}](t) = (\mathsf{Lin}[\mathsf{iv}](t))^{2^{e_*}} + \mathsf{ct} \cdot \mathsf{Lin}[\mathsf{iv}](t)$$

where t_i, $i = 1, 2, \ldots, \ell$, is the output of the i-th S-box and $t \stackrel{\text{def}}{=} [t_1|\ldots|t_\ell]$. Since every multiplication triple shares the same multiplier pt, a single value of α can be included in the signature instead of $\ell + 1$ different values.

Locally Computed Output Shares. For the above multiplication triples, every multiplication output share on the right-hand side can be locally computed without communication between parties. Hence, it is possible to remove the share Δz in the signature. This technique is similar with *multiplications with public output*, suggested in BN++.

For the first ℓ multiplications, each party computes the output as $(\mathsf{pt}^{(i)})^{2^{e_i}}$ based on their input share $\mathsf{pt}^{(i)}$ using linear operations. For the last multiplication output, the output is determined as follows.

$$\begin{cases} (A_{\mathsf{iv}} \cdot t^{(i)} + b_{\mathsf{iv}})^{2^{e_*}} + \mathsf{ct} \cdot (A_{\mathsf{iv}} \cdot t^{(i)} + b_{\mathsf{iv}}) & \text{for } i = 1, \\ (A_{\mathsf{iv}} \cdot t^{(i)})^{2^{e_*}} + \mathsf{ct} \cdot (A_{\mathsf{iv}} \cdot t^{(i)}) & \text{for } i \geq 2, \end{cases}$$

where $t^{(i)} \in \mathbb{F}_2^{\ell n}$ is the output shares of the first ℓ S-boxes for the i-th party: $t^{(i)} = [t_1^{(i)} | \dots | t_\ell^{(i)}]$.

With the above optimization techniques applied, the signature size is given as

$$6\lambda + \tau \cdot (\lambda \cdot \lceil \log_2(N) \rceil + (\ell + 5) \cdot \lambda).$$

Other Symmetric Primitives In Use. The SHAKE128 (resp. SHAKE256) XOF is used to instantiate hash functions Commit, H_1, H_2 and pseudorandom generators Expand and $\mathsf{ExpandTape}$ in the signature scheme for $\lambda = 128$ (resp. $\lambda \in \{192, 256\}$). $\mathsf{Sample}(\mathsf{tape})$ samples an element from a random tape tape, which is an output of $\mathsf{ExpandTape}$, tracking the current position of the tape.

Algorithm 1: $\mathsf{Sign}(\mathsf{pt}, (\mathsf{iv}, \mathsf{ct}), m)$ - AIMer signature scheme, signing algorithm.

`// Phase 1: Committing to the seeds and the execution views of the parties.`

Sample a random salt $\mathsf{salt} \xleftarrow{\$} \{0,1\}^{2\lambda}$.
Compute the first ℓ S-boxes' outputs $t_1, \ldots, t_\ell$.
Derive the binary matrix $A_{\mathsf{iv}} \in (\mathbb{F}_2^{n \times n})^\ell$ and the vector $b_{\mathsf{iv}} \in \mathbb{F}_2^n$ from the initial vector iv.
for *each parallel execution* $k \in [\tau]$ **do**
  Sample a root seed : $\mathsf{seed}_k \xleftarrow{\$} \{0,1\}^\lambda$.
  Compute parties' seeds $\mathsf{seed}_k^{(1)}, \ldots, \mathsf{seed}_k^{(N)}$ as leaves of binary tree from seed_k.
  for *each party* $i \in [N]$ **do**
    Commit to seed: $\mathsf{com}_k^{(i)} \leftarrow \mathsf{Commit}(\mathsf{salt}, k, i, \mathsf{seed}_k^{(i)})$.
    Expand random tape: $\mathsf{tape}_k^{(i)} \leftarrow \mathsf{ExpandTape}(\mathsf{salt}, k, i, \mathsf{seed}_k^{(i)})$.
    Sample witness share: $\mathsf{pt}_k^{(i)} \leftarrow \mathsf{Sample}(\mathsf{tape}_k^{(i)})$.
  Compute witness offset and adjust first witness: $\Delta\mathsf{pt}_k \leftarrow \mathsf{pt} - \sum_i \mathsf{pt}_k^{(i)}$, $\mathsf{pt}_k^{(1)} \leftarrow \mathsf{pt}_k^{(1)} + \Delta\mathsf{pt}_k$.
  for *each S-box with index* j **do**
    if $j \leq \ell$ **then**
      For each party i, sample a S-box output: $t_{k,j}^{(i)} \leftarrow \mathsf{Sample}(\mathsf{tape}_k^{(i)})$.
      Compute output offset and adjust first share: $\Delta t_{k,j} = t_j - \sum_i t_{k,j}^{(i)}$, $t_{k,j}^{(1)} \leftarrow t_{k,j}^{(1)} + \Delta t_{k,j}$.
      For each party i, set $x_{k,j}^{(i)} = t_{k,j}^{(i)}$ and $z_{k,j}^{(i)} = (\mathsf{pt}_k^{(i)})^{2^{e_j}}$.
    if $j = \ell + 1$ **then**
      For $i = 1$, set $x_{k,j}^{(i)} = A_{\mathsf{iv}} \cdot t_{k,*}^{(i)} + b_{\mathsf{iv}}$ where $t_{k,*}^{(i)} = [t_{k,1}^{(i)} | \ldots | t_{k,\ell}^{(i)}]$ is the output shares of the first ℓ S-boxes.
      For each party $i \in [N] \backslash \{1\}$, set $x_{k,j}^{(i)} = A_{\mathsf{iv}} \cdot t_{k,*}^{(i)}$
      For each party i, set $z_{k,j}^{(i)} = (x_{k,j}^{(i)})^{2^{e_*}} + \mathsf{ct} \cdot x_{k,j}^{(i)}$.
  For each party i, set $a_k^{(i)} \leftarrow \mathsf{Sample}(\mathsf{tape}_k^{(i)})$.
  Compute $a_k = \sum_{i=1}^N a_k^{(i)}$.
  Set $c_k = a_k \cdot \mathsf{pt}$.
  For each party i, set $c_k^{(i)} \leftarrow \mathsf{Sample}(\mathsf{tape}_k^{(i)})$.
  Compute offset and adjust first share : $\Delta c_k = c_k - \sum_i c_k^{(i)}$, $c_k^{(1)} \leftarrow c_k^{(1)} + \Delta c_k$.
Set $\sigma_1 \leftarrow \Big(\mathsf{salt}, ((\mathsf{com}_k^{(i)})_{i \in [N]}, \Delta\mathsf{pt}_k, \Delta c_k, (\Delta t_{k,j})_{j \in [\ell]})_{k \in [\tau]}\Big)$.

`// Phase 2: Challenging the checking protocol.`

Compute challenge hash: $h_1 \leftarrow H_1(m, \mathsf{iv}, \mathsf{ct}, \sigma_1)$.
Expand hash: $((\epsilon_{k,j})_{j \in [\ell+1]})_{k \in [\tau]} \leftarrow \mathsf{Expand}(h_1)$ where $\epsilon_{k,j} \in \mathbb{F}_{2^n}$.

`// Phase 3. Commit to the simulation of the checking protocol.`

for *each repetition* k **do**
  Simulate the triple checking protocol as in Section 2.3 for all parties with challenge $\epsilon_{k,j}$. The inputs are $((x_{k,j}^{(i)}, \mathsf{pt}_k^{(i)}, z_{k,j}^{(i)})_{j \in [\ell+1]}, a_k^{(i)}, b_k^{(i)}, c_k^{(i)})$, where $b_k^{(i)} = \mathsf{pt}_k^{(i)}$, and let $\alpha_k^{(i)}$ and $v_k^{(i)}$ be the broadcast values.
Set $\sigma_2 \leftarrow \Big(\mathsf{salt}, ((\alpha_k^{(i)}, v_k^{(i)})_{i \in [N]})_{k \in [\tau]}\Big)$.

`// Phase 4. Challenging the views of the MPC protocol.`

Compute challenge hash: $h_2 \leftarrow H_2(h_1, \sigma_2)$.
Expand hash: $(\bar{i}_k)_{k \in [\tau]} \leftarrow \mathsf{Expand}(h_2)$ where $\bar{i}_k \in [N]$.

`// Phase 5. Opening the views of the MPC and checking protocols.`

for *each repetition* k **do**
  $\mathsf{seeds}_k \leftarrow \{\lceil \log_2(N) \rceil$ nodes to compute $\mathsf{seed}_k^{(i)}$ for $i \in [N] \backslash \{\bar{i}_k\}\}$.
Output $\sigma \leftarrow (\mathsf{salt}, h_1, h_2, (\mathsf{seeds}_k, \mathsf{com}_k^{(\bar{i}_k)}, \Delta\mathsf{pt}_k, \Delta c_k, (\Delta t_{k,j})_{j \in [\ell]}, \alpha_k^{(\bar{i}_k)})_{k \in [\tau]})$.

Algorithm 2: $\mathsf{Verify}((\mathsf{iv}, \mathsf{ct}), m, \sigma)$ - AIMer signature scheme, verification algorithm.

Parse σ as $\left(\mathsf{salt}, h_1, h_2, \left(\mathsf{seeds}_k, \mathsf{com}_k^{(\bar{i}_k)}, \Delta\mathsf{pt}_k, \Delta c_k, (\Delta t_{k,j})_{j\in[\ell]}, \alpha_k^{(\bar{i}_k)}\right)_{k\in[\tau]}\right)$.

Derive the binary matrix $A_{\mathsf{iv}} \in (\mathbb{F}_2^{n\times n})^\ell$ and the vector $b_{\mathsf{iv}} \in \mathbb{F}_2^n$ from the initial vector iv.

Expand hashes: $((\epsilon_{k,j})_{j\in[\ell+1]})_{k\in[\tau]} \leftarrow \mathsf{Expand}(h_1)$ and $(\bar{i}_k)_{k\in[\tau]} \leftarrow \mathsf{Expand}(h_2)$.

for *each parallel repetition* $k \in [\tau]$ **do**

- Uses seeds_k to recompute $\mathsf{seed}_k^{(i)}$ for $i \in [N] \setminus \{\bar{i}_k\}$.
- **for** *each party* $i \in [N] \setminus \{\bar{i}_k\}$ **do**
  - Recompute $\mathsf{com}_k^{(i)} \leftarrow \mathsf{Commit}(\mathsf{salt}, k, i, \mathsf{seed}_k^{(i)})$,
  - $\mathsf{tape}_k^{(i)} \leftarrow \mathsf{ExpandTape}(\mathsf{salt}, k, i, \mathsf{seed}_k^{(i)})$ and
  - $\mathsf{pt}_k^{(i)} \leftarrow \mathsf{Sample}(\mathsf{tape}_k^{(i)})$.
  - **if** $i = 1$ **then**
    - Adjust first share: $\mathsf{pt}_k^{(i)} \leftarrow \mathsf{pt}_k^{(i)} + \Delta\mathsf{pt}_k$
  - **for** *each S-box with index* j **do**
    - **if** $j \le \ell$ **then**
      - Sample a S-box output: $t_{k,j}^{(i)} \leftarrow \mathsf{Sample}(\mathsf{tape}_k^{(i)})$.
      - **if** $i = 1$ **then**
        - Adjust first share: $t_{k,j}^{(1)} \leftarrow t_{k,j}^{(1)} + \Delta t_{k,j}$.
      - Set $x_{k,j}^{(i)} = t_{k,j}^{(i)}$ and $z_{k,j}^{(i)} = (\mathsf{pt}_k^{(i)})^{2^{e_j}}$.
    - **if** $j = \ell + 1$ **then**
      - **if** $i = 1$ **then**
        - Set $x_{k,j}^{(i)} = A_{\mathsf{iv}} \cdot t_{k,*}^{(i)} + b_{\mathsf{iv}}$ where $t_{k,*}^{(i)} = [t_{k,1}^{(i)} | \dots | t_{k,\ell}^{(i)}]$ is the output shares of the first ℓ S-boxes.
      - **else**
        - Set $x_{k,j}^{(i)} = A_{\mathsf{iv}} \cdot t_{k,*}^{(i)}$.
      - Set $z_{k,j}^{(i)} = (x_{k,j}^{(i)})^{2^{e_*}} + \mathsf{ct} \cdot x_{k,j}^{(i)}$.
  - Set $a_k^{(i)} \leftarrow \mathsf{Sample}(\mathsf{tape}_k^{(i)})$ and $c_k^{(i)} \leftarrow \mathsf{Sample}(\mathsf{tape}_k^{(i)})$.
  - **if** $i = 1$ **then**
    - Adjust first share $c_k^{(i)} \leftarrow c_k^{(i)} + \Delta c_k$.

Set $\sigma_1 \leftarrow \left(\mathsf{salt}, \left((\mathsf{com}_k^{(i)})_{i\in[N]}, \Delta\mathsf{pt}_k, \Delta c_k, (\Delta t_{k,j})_{j\in[\ell]}\right)_{k\in[\tau]}\right)$.

Set $h_1' \leftarrow H_1(m, \mathsf{iv}, \mathsf{ct}, \sigma_1)$.

for *each parallel execution* $k \in [\tau]$ **do**

- **for** *each party* $i \in [N] \setminus \{\bar{i}_k\}$ **do**
  - Simulate the triple checking protocol as defined in Section 2.3 for all parties with challenge $\epsilon_{k,j}$. The inputs are $((x_{k,j}^{(i)}, \mathsf{pt}_k^{(i)}, z_{k,j}^{(i)})_{j\in[\ell+1]}, a_k^{(i)}, b_k^{(i)}, c_k^{(i)})$, where $b_k^{(i)} = \mathsf{pt}_k^{(i)}$, and let $\alpha_k^{(i)}$ and $v_k^{(i)}$ be the broadcast values.

Compute $v_k^{(\bar{i}_k)} = 0 - \sum_{i\ne\bar{i}_k} v_k^{(i)}$.

Set $\sigma_2 \leftarrow \left(\mathsf{salt}, ((\alpha_k^{(i)}, v_k^{(i)})_{i\in[N]})_{k\in[\tau]}\right)$

Set $h_2' = H_2(h_1, \sigma_2)$.

Output Accept if $h_1 = h_1'$ and $h_2 = h_2'$.

Otherwise, output Reject.