# Algebraic Cryptanalysis of HADES Design Strategy: Application to POSEIDON and Poseidon2

Tomer Ashur
*Polygon Research*
*Cryptomeria, Leuven, Belgium*
tomer@cryptomeria.tech

Thomas Buschman
*Eindhoven University of Technology*
t.buschman@student.tue.nl

Mohammad Mahzoun
*Eindhoven University of Technology*
m.mahzoun@tue.nl

## Abstract

The HADES design strategy aims to provide an efficient way to instantiate Arithmetization-Oriented primitives by generalizing substitution-permutation networks to include partial S-box rounds. A notable instance of HADES, introduced by Grassi *et al.* at USENIX Security '21, is POSEIDON. Owing to its impressive efficiency and low arithmetic complexity, Poseidon has garnered attention from designers of integrity-proof systems. An updated version of POSEIDON, namely, Poseidon2 was published recently at AfricaCrypt '23 aiming to improve the efficiency of POSEIDON by optimizing its linear operations.

In this work, we show some caveats in the security argument of HADES against algebraic attacks. We provide an upper bound on the complexity of XL attacks against the HADES instances POSEIDON and Poseidon2. When the desired security level is high, some instances of these hash functions fail to provide the promised security. In particular, the complexity of the XL attack against an instance of POSEIDON and Poseidon2 claiming 512 bits of desired security is upper bounded by 402.64 bits.

Furthermore, we quantify the complexity of Gröbner basis attacks as a function of the number of S-boxes. We observe that the complexity is lower than claimed with the direct implication that there are cases where the recommended number of rounds is insufficient for meeting the claimed security. Concretely, the complexity of a Gröbner basis attack for an instance with 1024 bits of security is 731.77 bits and the original security argument starts failing already at the 384-bit security level.

The findings presented in this paper are asymptotic in nature and at this moment, only non-standard security levels seem to be practically affected. The results were shared with the designers.

## 1 Introduction

Arithmetization-Oriented (AO) primitives are a common building block for advanced cryptographic protocols such as Zero-Knowledge (ZK) proofs, Multiparty Computation (MPC), and Fully Homomorphic Encryption (FHE). AO primitives are usually defined over a finite field of large order and designed to have a simple and efficient algebraic representation. Examples of such primitives are *Rescue* [4], *Rescue-Prime* [42], *RPO* [5], and *Chaghri* [6] which are designed based on the Marvellous design strategy [4], Griffin [28] and Anemoi [14] which are Feistel-like designs, POSEIDON [29] and Poseidon2 [30] based on HADES design strategy, and more examples such as *MiMC* [2], *LowMC* [3], *Kreyvium* [16], *FLIP* [39], *Rasta* [23], *Dasta* [33], *Pasta* [24], *Fasta* [18], *Elisabeth* [19], *Rubato* [32], *Tip5* [43], to name just a few.

A promising approach for designing efficient AO primitives is the HADES design strategy. POSEIDON and its successor Poseidon2 are the most important and widely used hash functions designed based on HADES approach. POSEIDON is an efficient hash function operating over a prime field $\mathbb{F}_p$. It is a sponge function [10] instantiated by the POSEIDON$^\pi$ permutation. Recently, an optimized version, Poseidon2, was proposed which only differs in the underlying permutation.

The main idea of the HADES design strategy is to find the ideal combination of substitution-permutation network (SPN) rounds called "full layers" and a Partial SPN [27] (PSPN) rounds called "partial layers" to ensure efficiency. In HADES, full layers are used to justify arguments for the primitive's resistance against statistical attacks using the wide trail strategy. Then, partial rounds are efficient rounds that not only improve the performance of the design but also in combination with full layers ensure resistance against algebraic attack. Indeed, it is claimed that both full and partial layers provide ***same*** resistance in case of algebraic attacks [31]. The flexibility in utilizing the SPN and PSPN rounds provides the designers the opportunity to design optimal primitives for various use cases. The security of the HADES approach is based on an extensive analysis of various techniques such as:

- Statistical attacks: Differential cryptanalysis, linear cryptanalysis.
- Algebraic attacks: Interpolations attacks [34], Gröbner basis attacks [21], Higher-Order differential attacks [36], and Zero-Sum partitions attacks [13].

The security arguments of HADES and its instances were scrutinized by third-party cryptanalysts which presented security vulnerabilities exploiting the partial layers [12, 35]. To wit, it has been observed that under certain conditions, the linear operation of partial layers results in invariant subspaces. Subsequently, the security arguments and suggested secure parameters were updated accordingly by imposing additional constraints on the choice of the linear layer in the partial layer. Later, [9] showed how to bypass two full rounds as an auxiliary approach for mounting algebraic attacks but no parameter sets were yet shown to be vulnerable. Sauer designed an algebraic attack to POSEIDON [40] and showed that the resistance of POSEIDON against Gröbner basis attacks is overstated. However, he did not provide any instance that is indeed vulnerable.

**Our contributions**. We investigate the feasibility of Gröbner basis attacks and XL attacks against POSEIDON and Poseidon2 as the most important instances of HADES. Our approach is to conduct a thorough security analysis against the CICO problem in the context of Gröber basis and XL attacks, which are considered to be among the most promising algebraic attacks against AO designs [1].

To perform XL cryptanalysis, we calculate the smallest degree $D$ such that when the polynomial system describing POSEIDON is extended to degree $D$, the system is over-determined and can be solved using linear algebra techniques. An example of an instance whose claimed security is violated using the XL attack can be found in Table 1

| $\lambda$ | $\log_2(p)$ | $\alpha$ | $t$ | $r$ | $R_F$ | $R_P$ | $\mathcal{C}_{\text{XL}}$ |
|---|---|---|---|---|---|---|---|
| 512 | 64 | 3 | 24 | 8 | 8 | 42 | **402.64** |

Table 1: An instance of POSEIDON and Poseidon2 hash functions with security parameter $\lambda$ over the finite field $\mathbb{F}_p$. $\mathcal{C}_{\text{XL}}$ is the upper bound of the complexity of the XL attack.

To compute an **upper bound** for the complexity of Gröbner basis attacks, we use the state-of-the-art approach to compute the solving degree through extrapolation and calculate an upper bound for it. Extrapolation of the solving degree is considered to be the most accurate estimation of the Gröbner basis complexity [4, 5, 9, 14, 42, 43]. Using our upper bound, we show that the number of rounds is insufficient to provide the claimed security level when the security level is high. There are two main reasons for the overestimation of the security of the POSEIDON instances. Firstly, the degree of regularity of the system was assumed to follow the Macaulay bound [37], which was shown to be inaccurate [40] and quantified in our work. Secondly, it was believed that partial rounds provide the same security resistance against algebraic attacks as full rounds. We demonstrate that in cases where the state size is larger than two, partial rounds offer significantly less resistance against Gröbner basis attacks than full rounds. To show the impact of our observations and as a proof of concept, we demonstrate an instance with 1024 bits of security which are broken by our approach; the complete parameter set can be found in Table 2.

| $\lambda$ | $\log_2(p)$ | $\alpha$ | $t$ | $r$ | $R_F$ | $R_P$ | $\mathcal{C}_{\text{GB}}$ |
|---|---|---|---|---|---|---|---|
| 1024 | 128 | 3 | 24 | 8 | 8 | 85 | **731.77** |

Table 2: an instance of POSEIDON hash function with security parameter $\lambda$, state size $t$, rate $r$, and $(R_F, R_P)$ the number of full and partial rounds, respectively. $\mathcal{C}_{\text{GB}}$ is the complexity of the Gröbner basis attack.

The proposed algebraic attacks suggest that the partial layers do not provide the expected level of security against algebraic attacks, requiring that the security argument be re-evaluated for instances following this design strategy.

Additionally, we conducted a more thorough investigation into POSEIDON's security argument with respect to the claimed resistance against algebraic attacks. We revealed three distinct flaws in these arguments, each of which has implications for the required number of rounds. First, we show a typo in the security argument against Gröbner basis attacks in the full round setting. Then, we show that the logical reasoning for the security argument against the Gröbner basis attack is not sound. Finally, we present an error in the symbolic computation of bounds that undermines security.

**Structure of the Paper.** In Section 2, the notations used throughout the paper and the required background materials are described. In Section 3, an overview of POSEIDON and Poseidon2 designs, their security arguments, and the flaws in the security arguments are outlined. In Section 4, a Gröbner basis attack is proposed, and vulnerable instances are demonstrated. In Section 5, the XL attack is described and broken instances are described. Finally, in Section 6, the paper is summarized, the steps taken toward disclosure are outlined, and possible directions for future research are discussed.

# 2 Preliminaries

## 2.1 Notations

In this paper, we define $\lambda$ as the security parameter. To show an inclusive range of numbers, we use $[a,b] = \{a,\ldots,b\}$. Vectors are denoted by bold capital letters such as $\mathbf{X},\mathbf{Y},\mathbf{Z},\ldots$ and the elements of the vector $\mathbf{X}$ are denoted by $(x_1,\ldots,x_n)$. Matrices are denoted by calligraphic capital letters such as $\mathcal{M},\mathcal{N}$ where $\mathcal{M}_{i,j}$ is the $j^{th}$ element in the $i^{th}$ row.

**Definition 2.1** (Macaulay Matrix [38]). Let $\mathcal{P} \in K[x_1,\ldots,x_n]$ be a polynomial system with monomial ordering $\prec$, the Macaulay matrix $\mathcal{M}[d](\mathcal{P})$ of degree $d$ is a matrix with coefficients in $K$, where $\mathcal{M}[d]_{i,j}$ is the coefficient of the $j^{\text{th}}$ biggest monomial with respect to $\prec$ in the $i^{\text{th}}$ polynomial in the extended system. For example, let $\mathcal{P} = \{P_1,P_2\} = \{x^2+xy,y\}$. Then $\mathcal{M}[2](\mathcal{P})$ for *degrevlex* order is defined as:

$$\mathcal{M}[2](\mathcal{P}) = \begin{array}{cc} \begin{array}{cccccc} x^2 & xy & y^2 & x & y & 1 \end{array} & \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix} & \begin{array}{c} P_1 \\ P_2 \\ xP_2 \\ yP_2 \end{array} \end{array} .$$

**Definition 2.2** (Linear Algebra Constant ($\omega$) [44]). In the rest of this paper, $2 < \omega \leq 2.3727$ is defined as the linear algebra constant and is the complexity of matrix multiplication.

## 2.2 Polynomial Systems and How to Solve Them

Let $K$ be a field, the polynomial ring $K[x_1,\ldots,x_n]$ is a set of all polynomials in the variables $x_1,\ldots,x_n$ and coefficients in $K$. A polynomial system is a finite set of polynomials $P_1,\ldots,P_m \in K[x_1,\ldots,x_n]$ such that:

$$\begin{cases} P_1(x_1,\ldots,x_n) = 0 \\ P_2(x_1,\ldots,x_n) = 0 \\ \qquad \vdots \\ P_m(x_1,\ldots,x_n) = 0 \end{cases}$$

The polynomial systems typically describing AO hash functions span a zero-dimensional ideal, meaning that the set containing all their solutions is finite. In general, solving a polynomial system is considered to be NP-hard except in particular cases, *e.g.*, when $P_1,\ldots,P_m$ are linear functions. For the case when $P_1,\ldots,P_m$ are not linear generic methods exist. In Sections 2.2.1 and 2.2.2 we discuss the best currently known methods to solve a general system. As a word of caution we stress that *ad hoc* methods may outperform these generic algorithms in particular cases, hence their complexity should be interpreted only as an upper bound.

### 2.2.1 Univariate Polynomial Systems

The case where $n = 1$ is called a univariate polynomial system. When solving a univariate polynomial system of degree $D$ defined over the finite field $\mathbb{F}_p$ the Cantor/Zassenhaus [17] algorithm can be used with complexity [41]:

$$O(D^2(\log D \log\log D)(\log p + \log D)).$$

### 2.2.2 Multivariate Polynomial Systems

When $n > 1$ the polynomial system is said to be multivariate. To solve multivariate polynomial systems defined over a finite field, Gröbner basis methods are often used. The steps to solve a multivariate polynomial system using one of the Gröbner basis algorithms are:

1. Compute a Gröbner basis with respect to *degrevlex* term order.
2. Convert the Gröbner basis to *lex* term order.
3. Find the roots of the polynomial system by factoring univariate polynomials and extending the partial solutions.

The primary motivation for first computing the Gröbner basis in *degrevlex* order is its lower complexity compared to other term orderings.

**Complexity of Step 1.** The complexity of computing a Gröbner basis in *degrevlex* term order is upper bounded by [11]:

$$O\left(\binom{n+d_{sol}}{d_{sol}}^{\omega}\right), \tag{1}$$

where $n$ is the number of variables in the multivariate polynomial system and $d_{sol}$ is the solving degree of the polynomial system [22].

**Solving Degree.** There are multiple notions of degrees in the literature that aim to capture the complexity of the Gröbner basis computation [8, 15]. In this work, we use the *solving degree*, which is defined as the highest degree of the polynomials involved in the computation of the Gröbner basis using the celebrated F5 algorithm. In the case of regular systems, the solving degree matches the Macaulay bound which is defined as:

$$d_{sol} = \sum_{i=1}^{m}(d_i - 1) + 1,$$

where $d_i$ is the polynomial degree of $P_i$ for $1 \leq i \leq m$.

However, most of the AO primitives, when modeled as a polynomial system, are not regular systems [4,9,40], and in most of the cases, the solving degree grows slower than the Macaulay bound.

To determine the solving degree, the current state-of-the-art approach used in design and cryptanalysis is to compute the solving degrees of round-reduced versions of the system, and extrapolate a bound for it [1,4,9,14,40].

**Complexity of Step 2.** The computed Gröbner basis in *degrevlex* order is usually complicated and not useful for solving the system. Therefore, it is converted to a Gröbner basis in *lex* order. The conversion is performed using the FGLM [25] algorithm and the complexity is upper bounded by:

$$O\left(nD^3\right),$$

where $D$ is the degree of the zero-dimensional ideal. In some cases, this step can be performed more efficiently using the sparse FGLM [26] algorithm which has asymptotic complexity of:

$$O(\sqrt{6/n\pi}D^{2+\frac{n-1}{n}}). \tag{2}$$

**Complexity of Step 3.** When the ideal is zero-dimensional (as is the case for us), the Gröbner basis in the lex order contains a unique univariate polynomial that can be factored and is used to iteratively solve the entire system. When the unique univariate polynomial is factored, it results in a partial solution to the system. In an iterative process, partial solutions are substituted in other polynomials, and these are factored in a similar way until a full solution is obtained. The complexity of factoring a univariate polynomial is described in Section 2.2.1.

### 2.2.3 The XL Attack

Another approach to solving multivariate polynomial systems is the family of *eXtended Linearization* (XL) [20] algorithms. To describe how the XL attack works, we first state the Lemma 2.1.

**Lemma 2.1.** The number of monomials in variables $x_1, \ldots, x_n$ with degree less than or equal to $D$ is $\binom{n+D}{D-1}$.

*Proof.* Let the degree of $x_i$ be $a_i$, then the degree of the monomial is $a_1 + \ldots + a_n$. The degree is less than or equal to $D$ if:

$$a_1 + \ldots + a_n \leq D \iff a_1 + \ldots + a_n + b = D,$$

where each $a_i$ for $1 \leq i \leq n$ and $b$ can take any value from 0 to $D$. The number of solutions for such a system is:

$$\binom{n+D}{D-1}.$$

. □

The core idea of XL is to extend a polynomial system by multiplying its polynomials by all monomials up to a certain degree. More precisely, the polynomial system $\mathcal{P} = \{P_1, \ldots, P_m\}$ in variables $x_1, \ldots, x_n$ is extended to:

$$\mathcal{P}_{\text{ext}}[d] = \{m \cdot P_i | m \in \mathcal{M}_d\},$$

where $\mathcal{M}_d$ is a set of all monomials of degree at most $d$ in the same variables and is of size $\binom{d+n}{d-1}$. The extended system $\mathcal{P}_{\text{ext}}[d]$ contains $m\binom{d+n}{d-1}$ polynomials. Consider the case that all $P_i$ have the same degree $\alpha$, then the maximum degree of $\mathcal{P}_{\text{ext}}[d]$ is $d+\alpha$ and number of monomials in $\mathcal{P}_{\text{ext}}[d]$ is $\binom{d+\alpha+n}{d+\alpha-1}$. When the number of monomials is not larger than the number of polynomials in the system, the Macaulay matrix of the polynomial system is over-determined and can be solved using linear algebra techniques. The complexity of the XL attack is upper bounded by:

$$O\left(\binom{d+\alpha+n}{d+\alpha-1}\right)^{\omega}. \tag{3}$$

## 2.3 The Sponge Construction

The Sponge construction [10] is a mode of operation that transforms a fixed-length permutation to a hash function, or in general a sponge function, that has variable-length inputs and outputs. Let $\mathbb{F}_p$ be a finite field of order $p$ and $f : \mathbb{F}_p^n \to \mathbb{F}_p^n$ be a fixed-length transformation operating over a state of size $n$ with elements in $\mathbb{F}_p$. The sponge function $F$ with rate $r$ and capacity $c$ where $r + c = n$, takes as input $\mathbf{M}$ of arbitrary length, and after applying a padding function, generates the output $\mathbf{H}$. The length of the padded input and the output of $F$ is a multiple of $r$. The sponge function works as follows:

1. Let $S$ be the state of the sponge function of length $n = r + c$.
2. The state $\mathbf{S}$ is initialized to $(0, \ldots, 0)$.
3. Absorbing phase: The padded message $\mathbf{M}$ is split into $\chi$ blocks $\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_\chi$ of length $r$. For each $i \in 1, 2, \ldots, \chi$, the $\mathbf{M}_i$ is added to the first $r$ blocks of $\mathbf{S}$ and the function $f$ is applied *i.e.*,

$$\mathbf{S} = f(\mathbf{S} + \mathbf{M}_i)$$

4. Squeezing phase: once all blocks of the padded message have been absorbed, the squeezing phase starts to generate the output. In this phase, the function outputs blocks $\mathbf{H}_1, \ldots, \mathbf{H}_{\chi'}$ of length $r$ and update the internal state $\mathbf{S}$ by applying the function $f$.

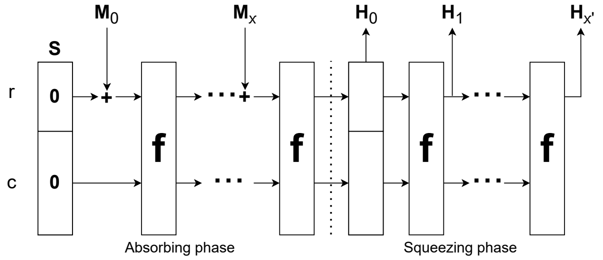

Figure 1: A sponge function with rate $r$, capacity $c$, and internal permutation $f$.

In Figure 1, a schematic construction of the sponge function is illustrated. To study more about sponge construction in the context of ZK-friendly hash functions, we refer to [7].

Assuming that $f$ is computationally indistinguishable from a random permutation, a sponge function with capacity $c$ offers $2^{c/2}$ bits of collision resistance and preimage resistance [10].

## 2.4 Security Definition of AO Hash Functions

AO hash functions need to provide security against preimage and collision attacks. Additionally, specific AO hash functions, such as POSEIDON and Poseidon2 are required to be secure against Constrained Input-Constraint Output (CICO) problem.

**Definition 2.3** (Preimage resistance)**.** A hash function $H : D \to R$ is preimage resistant if for any $y \in R$ it is computationally infeasible to find $x \in D$ such that $H(x) = y$ except in cases where $x$ was already queried to $H(\cdot)$.

**Definition 2.4** (Second-preimage resistance)**.** A hash function $H : D \to R$ is second-preimage resistant if for a given $x \in D$, it is computationally infeasible to find $x' \in D$ such that $H(x) = H(x')$.

**Definition 2.5** (Collision resistance)**.** A hash function $H : D \to R$ is collision resistant if it is computationally infeasible to find $x, x' \in D$ such that $H(x) = H(x')$.

**Definition 2.6** (CICO resistance)**.** A hash function $H : D \to R$ is CICO resistant if it is computationally infeasible to find $x \| x' \in D$ and $y \| y' \in R$ such that $H(x \| x') = y \| y'$ where $\|$ represents the concatenation of elements.

## 2.5 The HADES Design Strategy

The HADES design strategy is a paradigm for developing efficient and secure AO primitives. HADES uses two types of SPN networks, known as *full layers*—placed at the beginning and at the end of the permutation—and *partial layers*, placed in the middle. Each full round of HADES works as follows:

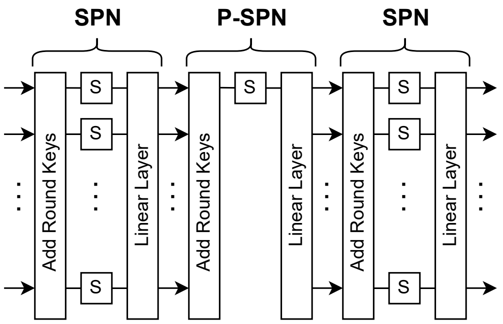

Figure 2: The SPN layer at the beginning and the end are full layers. The PSPN in the middle is the partial layer where the S-box only applies to the first element of the state.

1. Add round keys.
2. Substitution (non-linear) layer applied to all the elements in the state.
3. Permutation (linear) layer.

Each partial round of HADES works as follows:

1. Add round keys.
2. Substitution (non-linear) layer applied to specific elements, usually the first one, in the state.
3. Permutation (linear) layer.

In Figure 2 an overview of the HADES design strategy is depicted.

# 3 POSEIDON and Poseidon2

Let us denote the set of vectors over the finite field $\mathbb{F}_p$ with arbitrary length with $\mathbb{F}_p^*$. POSEIDON: $\mathbb{F}_p^* \to (\mathbb{F}_p^r)^{\chi'}$ is a hash function operating over $\mathbb{F}_p$ with output of $\chi'$ blocks of length $r$. It is constructed by using the POSEIDON$^\pi$ permutation in the sponge construction with rate $r$ and capacity $c$. POSEIDON$^\pi$ is a permutation with a state size of $t$ and consists of $R = R_F + R_P$ rounds, where $R_F = R_f + R_f$ rounds are full rounds with $t$ S-boxes, and $R_P$ rounds are partial rounds with only one S-box applied to the first element of the state. The POSEIDON$^\pi$ permutation is illustrated in Figure 3 and works as follows:

1. Add round constants: $ARC_{\mathbf{C}} : \mathbb{F}_p^t \to \mathbb{F}_p^t$, $ARC_{\mathbf{C}}(\mathbf{X}) = \mathbf{X} + \mathbf{C}$.

2. Substitution layer: $S_\alpha : \mathbb{F}_p \to \mathbb{F}_p$, $S_\alpha(x) = x^\alpha$, where the S-box is applied to the first element (in partial rounds), or all elements of the state (in full rounds).
3. Linear layer: $L_\mathcal{M} : \mathbb{F}_p^t \to \mathbb{F}_p^t$, $L_\mathcal{M}(\mathbf{X}) = \mathcal{M} \cdot \mathbf{X}^\intercal$ where $\mathcal{M}$ is a MDS matrix.

Where $\mathcal{M}$ is a Cauchy matrix [45] and is defined as follows:

$$\mathcal{M}_{i,j} = \frac{1}{x_i + y_j},$$

for pairwise distinct $x_i$ and $y_j$ with the condition that $x_i + y_j \neq 0$ for all $i, j \in [1, t]$.

To improve the efficiency of POSEIDON, the authors designed Poseidon2. Poseidon2 uses different partial rounds and different linear layers. Posedion2 works as follows:

1. Initial linear layer: $L_{\mathcal{M}'} : \mathbb{F}_p^t \to \mathbb{F}_p^t$, $L_{\mathcal{M}'}(\mathbf{X}) = \mathcal{M}' \cdot \mathbf{X}^\intercal$ where $\mathcal{M}'$ is an MDS matrix.
2. For $R$ rounds:
   (a) Add round constants: $ARC_\mathbf{C} : \mathbb{F}_p^t \to \mathbb{F}_p^t$
   $$ARC_\mathbf{C}(\mathbf{X}) = \mathbf{X} + \mathbf{C}.$$
   In the case of partial rounds, $\mathbf{C} = (c_1, 0, \ldots, 0)$.
   (b) Substitution layer: $S_\alpha : \mathbb{F}_p \to \mathbb{F}_p$, $S_\alpha(x) = x^\alpha$, where the S-box is applied to the first element (in partial rounds), or all elements of the state (in full rounds).
   (c) Linear layer: $L_\mathcal{M} : \mathbb{F}_p \to \mathbb{F}_p$, $L_\mathcal{M}(\mathbf{X}) = \mathcal{M} \cdot \mathbf{X}^\intercal$.

Where $\mathcal{M}$ is an MDS matrix. In case of full rounds, $\mathcal{M} = \mathcal{M}'$ and is same as the MDS matrices defined for Griffin-$\pi$ [28]. In case of partial rounds, $\mathcal{M} = \mathcal{M}''$ is defined in [30, Section 5.2]. Figure 3 depicts how POSEIDON and Poseidon2 work and the updated operations in Poseidon2 are denoted by dashed lines.

Instances of POSEIDON and Poseidon2 that provide $\lambda$ bits of security, guarantee that any algorithm that finds collision or preimage requires a complexity of at least $2^\lambda$. In the case of the CICO problem, as long as finding $x \| x' \in D$ using exhaustive search is not possible with complexity less than $2^\lambda$, any algorithm that finds such $x \| x' \in D$ requires a complexity of at least $2^\lambda$.

## 3.1 Security Claims for POSEIDON and Poseidon2

The analysis of POSEIDON's security involves evaluating the system's vulnerability to two categories of attacks, namely statistical attacks and algebraic attacks. The authors established a constraint on the secure number of rounds $R = R_F + R_P$ and the desired security level $\lambda$ by ensuring that none of the attacks can be executed with a complexity of less than $\lambda$ steps, thereby ensuring the system's resilience against potential attacks. In addition to this, the authors incorporated a security margin to minimize the risk of any unpredicted weaknesses in the system. The security margins are:

1. Two additional full rounds ($+2R_F$), and
2. 7.5% of more partial rounds ($+7.5\% R_P$).

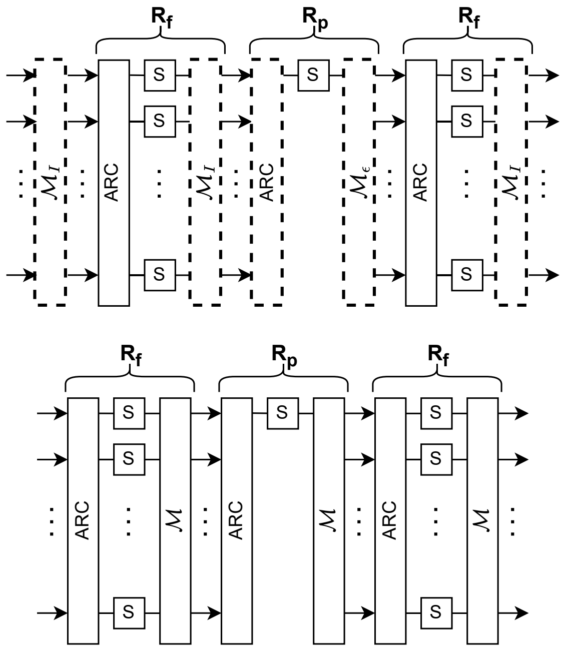

Figure 3: Construction of POSEIDON$^\pi$ (bottom), and POSEIDON2$^\pi$ (top) permutations. The modified steps are shown with dashed line

### 3.1.1 Generic Security

The sponge construction provides a generic security level of $2^{c/2}$ bits. In addition, an ideal hash function with security parameter $\lambda$ is expected to provide $2^\lambda$ bits of security against preimage attacks. To provide resistance against generic attacks, POSEIDON requires that $\lambda \leq \frac{c}{2}$ and $\lambda \leq r$.

## 3.2 Statistical Attacks

In [29, Equation 2], the minimum number of rounds to ensure security against statistical attacks for S-boxes of the form $S(x) = x^\alpha$ is described as:

$$\begin{cases} 6 & \text{if } \lambda \leq (\lfloor \log_2 p \rfloor - \log_2(\alpha - 1)) \cdot (t+1) \\ 10 & \text{otherwise.} \end{cases} \tag{4}$$

### 3.2.1 Algebraic Attacks

Evaluating the security of POSEIDON against algebraic attacks suggests that Interpolation attacks and Gröbner basis attacks have the lowest complexity. Therefore, the constraints on the number of rounds derived from these attacks are sufficient to provide resistance to other types of algebraic attacks.

**Interpolation Attacks.** In [29, Equation 3], it is asserted that for the security level of $\lambda$ bits, the maximum number of rounds vulnerable to the interpolation attack is:

$$R \leq \lceil \log_{\alpha}(2) . \min\{\lambda, \log_2(p)\} \rceil + \lceil \log_{\alpha}(t) \rceil \tag{5}$$

**Gröbner Basis Attacks.** In [29, Equation 5, 6], it is asserted that for the security level of $\lambda$ bits, the number of rounds vulnerable to Gröbner basis is:

$$\begin{cases} R \leq \log_{\alpha}(2) . \min\left\{\frac{\lambda}{3}, \frac{\log_2(p)}{2}\right\} \\ R \leq t - 1 + \min\left\{\frac{\log_{\alpha}(2) \cdot \lambda}{t+1}, \frac{\log_{\alpha}(2)\log_2(p)}{2}\right\} \end{cases} \tag{6}$$

To compute the complexity, it is presumed that any polynomial modelling of POSEIDON forms a regular sequence. In such a case, the the solving degree would coincide with the Macaulay bound [29, Section C.2.2].

## 3.3 Sufficient Number of Rounds

By applying the constraints specified in Equations (4)–(6) and the security margin, the total number of rounds to ensure the resistance of POSEIDON to studied attacks can be computed. A Python script is provided to facilitate the computation of the number of rounds[1]. In our analysis, we utilized this script to calculate the necessary number of rounds required to ensure the security of our chosen parameters.

## 3.4 Flaws in the Security Analysis of POSEIDON

In addition to the oversight described in Section 5 and the major flaw described in Section 4, we identified three more minor flaws which, when combined, increase the likelihood of an attack to exist. In Section 3.4.1, we demonstrate that using loose bounds in security arguments leads to incorrect conclusions. In Section 3.4.2, we investigate the security argument against Gröbner basis attacks in the case of $\chi = 1$, where the system is already a Gröbner basis in the full-permutation setting. We highlight a typo causing an underestimation of the required number of rounds. Finally, in Section 3.4.3, we identify a flaw in the symbolic computations of round-level Gröbner basis analysis that led to an overestimation of the number of rounds.

### 3.4.1 Improper Logic

The argument for determining the number of rounds that is safe against Gröbner basis attack can be summarized as follows [29, Section 5.5.2]:

1. Compute the complexity of the attack as a function of the POSEIDON parameters $\alpha, R_F, R_P, t, r, \chi, \lambda$.
2. Optionally, derive an upper bound for the computed complexity that is easier to manipulate.
3. Calculate the maximum number of rounds $R_F^*$ and $R_P^*$ that can be attacked given the parameters of POSEIDON.
4. Assume that all values for $R_F, R_P$ higher than $R_F^*, R_P^*$ cannot be attacked and are secure.

The problem arises due to Step 2, where a lower bound should be used. As a result, Step 4 concludes resistance against adversaries that Step 3 did not handle.

Consider a simple example: let us assume that a sponge construction with rate $r$ uses an $N$-round permutation; further, assume an attack with complexity $2^{3Nr}$. However, this expression may be challenging to work with (*e.g.*, because 3 is odd and we wish to take a square root) so we attempt to simplify it by noting that $2^{3Nr} \leq 2^{4Nr}$, although this is not a tight upper bound. Using the argumentation shown above, we find $N^*$ from:

$$2^{4N^*r} = 2^{\lambda}.$$

Solving for $N^*$ yields

$$N^* = \frac{\lambda}{4r}.$$

Consequently, for all $0 \leq N \leq N^*$, a sponge function using the $N$-round permutation can be attacked. This is still a true statement. Using the above argumentation, it is then conjectured that the sponge function is safe from attacks for all $N \geq N^* = \frac{\lambda}{4r}$. This is not a true statement as now using the proper expression to find a safe number of rounds $N_s$, we obtain

$$2^{3N_s r} = 2^{\lambda},$$

and find:

$$N_s = \frac{\lambda}{3r}.$$

Therefore, for all $0 \leq N \leq N_s$, the sponge function can be attacked, and for all $N > N_s$, the sponge function is safe for the given security level. The problem is that for $N^* \leq N \leq N_s$, we argued that the sponge construction with $N$ rounds is safe, while it is not the case. When using Step 3 and Step 4 outlined earlier, one should use

[1] https://extgit.iaik.tugraz.at/krypto/hadeshash

a lower bound rather than an upper bound in Step 2, as it may result in an overestimation of the resistance of the sponge function against attacks.

Similarly, the resistance of POSEIDON against a round-level Gröbner basis attack is found to be (up to reasonable approximation) [29]:

$$\mathcal{C}_{\mathrm{GB}} = 2^{Cq-C'},$$

with

$$C = 2\log_2\left(\frac{\alpha^\alpha}{(\alpha-1)^{\alpha-1}}\right)$$
$$C' = \log_2\left(\frac{2\pi(\alpha-1)q}{\alpha}\right)$$
$$q = (t-1)R_F + R_P + \chi$$

That concludes Step 1. In Step 2, this approximation was upper-bounded by:

$$\mathcal{C}_{\mathrm{GB}} = 2^{C\cdot q - C'} \leq 2^{C\cdot q}, \tag{7}$$

Ultimately, resistance against the round-level attack is assumed as long as:

$$(t-1)R_F + R_P \geq C^{-1}\min\{\lambda, \log_2(p)\} - 1, \tag{8}$$

Since (7) is not a tight bound (8) necessarily underestimates the required number of round. The effect of this omission is more noticeable when the power map $\alpha$, state size $t$, and rate $\chi$ grow.

### 3.4.2 Transcription Error

**Full-Permutation Equation**. In the full round equation setting [29, Section C.2.2], a system of equations for the entire $R$ rounds is derived by considering each input as a variable and applying the round functions to them. When the number of input variables $\chi$ is the same as the number of output variables, the resulting system will consist of $\chi$ equations in $\chi$ variables, and the degree of each polynomial is upper-bounded by $D_\alpha(R) = \alpha^R$.

When $\chi = 1$, the system consists of a single polynomial of degree at most $\alpha^R$ in one variable, which is already a Gröbner basis in lex order. Therefore, the only step required to complete the attack is the factorization of the univariate polynomial. Per the security argument provided in [29, Section C.2.2], one should have:

$$\log_2\left(\alpha^{\omega R}\right) \geq \log_2\left(\alpha^{2R}\right) \geq \min\{\lambda, \log_2(p)\},$$

which implies:

$$R \geq \left\lceil \frac{\min\{\lambda, \log_2(p)\}}{2\log_2\alpha} \right\rceil = \log_\alpha(2)\cdot\min\{\frac{\lambda}{\mathbf{2}}, \frac{\log_2(p)}{2}\},$$

where $R = R_F + R_P$. Later, the designers in [29, Equation 11], write the constraint for the full round attack as:

$$R_F + R_P \geq \log_\alpha(2)\cdot\min\{\frac{\lambda}{\mathbf{3}}, \frac{\log_2(p)}{2}\}, \tag{9}$$

where the denominator of the fraction $\frac{\lambda}{3}$ is 3 instead of 2. This mistake results in an overestimation of the security that the POSEIDON permutation provides against Gröbner basis attacks in the case where $\chi = 1$.

As an example of how the mistake influences the number of rounds, the constraint in [29, Equation 5] would imply that 6 full rounds and 22 partial rounds are sufficient for $\alpha = 3, t = 2, p \approx 2^{1024}$, and the desired security level of 128 bits, whereas to gain that security level for these parameters, at least 35 partial rounds are required.

### 3.4.3 Symbolic Computation Error

In [29, Section C.2.2] it is shown that for security level of $\lambda$, the maximum number of rounds that can be attacked using Gröbner basis is:

$$(t-1)R_F + R_P + \chi \leq C^{-1}\cdot\min\{\lambda, \log_2(p)\chi\}, \tag{10}$$

with

$$C = 2\log_2\left(\frac{\alpha^\alpha}{(\alpha-1)^{\alpha-1}}\right).$$

The designers argue that the maximal number of rounds that can be attacked is when $\chi = 1$ [29, Section C.2.2] but this is not true. Rewritting (10), we get

$$(t-1)R_F + R_P \leq C^{-1}\cdot\min\{\lambda - \chi C, \chi(\log_2(p) - C)\}.$$

Here, the first argument of the minimum function is indeed maximized for $\chi = 1$, but the last argument is maximized for $\chi = t-1$ because $\lambda - C$ is positive for the suggested parameters of POSEIDON. Ultimately, security is conjectured if:

$$(t-1)R_F + R_P \geq C^{-1}\cdot\min\{\lambda, \log_2(p)\} + t - 2,$$

but if we address the algebra error, we obtain:

$$(t-1)R_F + R_P \geq C^{-1}\cdot\min\{\lambda + C(t-2), \log_2(p)(t-1)\}.$$

Previously, the constraint for this kind of Gröbner basis attack appeared to be less restrictive than the other attacks, as it was subsumed by the constraints for the other kinds of Gröbner basis attacks [29, Equation 11]. However, once the error is addressed, this is no longer true. More importantly, there are parameter sets for which this constraint would require the highest number of partial rounds to be secure. For example, for $\alpha = 3, \log_2(p) \approx 256, \lambda = 1536, R_F = 8, t = 8$, an interpolation attack would be thwarted if $R_P \geq 158$, a subspace attack would fail if $R_P \geq 80$, and a full-permutation attack requires $R_P \geq 73$, *but* a round-level Gröbner basis attack require $R_P \geq 230$ to achieve required resistance. Therefore, [29, Equation. 5] requires three constraints rather than two and this omission does affect the required number of rounds for some parameter sets.

## 4 The Gröbner Basis Attack

The CICO resistance of POSEIDON and Poseidon2 is analyzed using the Gröbner basis attacks in this section. To solve CICO problem, we first model POSEIDON and Poseidon2 as a system of multivariate polynomials with known output and unknown input and we solve the system to find the desired input.

### 4.1 POSEIDON: Polynomial Modeling

POSEIDON is modeled for the case where $\alpha = 3$, the number of input blocks of size $r$ is $\chi = 1$ and the underlying permutation is applied only once. In POSEIDON, which is a sponge function, the first $r$ elements of the input state of the permutation are absorbed from the input, and the next $c$ elements are initialized to a constant value. Without loss of generality, we can assume that the last $c$ element of the input state is initialized to 0.

while it is possible to model POSEIDON in various ways using algebraic relations describing them, the model that minimizes the complexity of the Gröbner basis attack is the preferred one.

After a thorough analysis of various methods for polynomial modeling, we identified the approach used by Sauer [40] that aims to minimize the solving degree of the system results in the lowest theoretical complexity.

In the described polynomial system, $\mathbf{C}_i = \{c_{i,1}, \ldots, c_{i,t}\}$ denotes the round constants for the round $i \in \{1, \ldots, R\}$. $\mathbf{X}_i = \{x_{i,1}, \ldots, x_{i,t}\}$ are the variables that are describing the state of the round $i \in \{0, \ldots, R\}$, where $\mathbf{X}_0 = (x_1, \ldots, x_r, 0, \ldots, 0)$ is the input and $\mathbf{X}_R = (H_1, \ldots, H_r, x_{R,r+1}, \ldots, x_{R,t})$ is the output. The first round of the POSEIDON before multiplication by $\mathcal{M}$ can be described as:

$$x_{1,j} - (x_{0,j} + c_{1,j})^{\alpha} = 0 \qquad j \in [1, r],$$

that has $2r$ variables and $r$ polynomials. The state after the first and second S-box layers is modeled as follows:

$$x_{2,j} - \left( \left( \sum_{k=1}^{r} \mathcal{M}_{j,k} \cdot x_{1,k} + \sum_{k=r+1}^{t} \mathcal{M}_{j,k} \cdot c_{1,k}^{\alpha} \right) + c_{2,j} \right)^{\alpha} = 0,$$

where $j \in [1,t]$. The described polynomials add $t$ new variables and $t$ new polynomials to the system. The next $R_f$ full rounds (i.e., $3 \leq i \leq R_f$) are modeled as:

$$x_{i,j} - \left( \left( \sum_{k=1}^{t} \mathcal{M}_{j,k} \cdot x_{i-1,k} \right) + c_{i,j} \right)^{\alpha} = 0 \qquad j \in [1,t],$$

which add $(R_f - 2)t$ new variables and $(R_f - 2)$ new polynomials to the system. We introduce a variable $\mathbf{Y}$ to simplify the equations for partial rounds and it is initialized as:

$$\mathbf{Y}^{\intercal} = \mathcal{M} \cdot (x_{R_f,1}, \ldots, x_{R_f,t})^{\intercal}.$$

The partial rounds $R_f < i \leq R_f + R_P$ are modeled as:

$$\begin{aligned} & x_{i,1} - (y_1 + c_{i,1})^{\alpha} = 0 \\ & y_j = \mathcal{M}_{j,1} \cdot x_{i,1} + \sum_{k=2}^{t} \mathcal{M}_{j,k} \cdot (y_k + c_{i,k}) \qquad j \in [1,t], \end{aligned}$$

which add $R_P$ new variables and $R_P$ new polynomials to the system. The last $R_f$ rounds $R_f + R_p < i \leq R - 1$ are modeled as:

$$\begin{aligned} x_{i,j} - (y_j + c_{i,j})^{\alpha} &= 0 && j \in [1,t] \\ y_j &= \sum_{k=1}^{t} \mathcal{M}_{j,k} \cdot x_{i,k} && j \in [1,t], \end{aligned}$$

that add $(R_f - 1)t$ variables in $(R_f - 1)t$ polynomials to the system. Finally, the last round is modeled as:

$$\sum_{k=1}^{t} \mathcal{M}_{j,k}^{-1} \cdot x_{R,k} - (y_j + c_{R,j})^{\alpha} = 0 \qquad j \in [1,t].$$

The last round adds $c$ new variables and $t$ polynomial to the system. The final system has $r + (R_F - 1)t + R_P$ polynomials of degree $\alpha$ in $r + (R_F - 1)t + R_P$ variables.

Considering that $t$ and $r$ as constants, the linear regression of $d_{sol}$ on number of rounds $R_F, R_P$ based on the experimental data is:

$$d_{sol} = r\frac{R_F}{2} + 0.8R_P + \alpha \tag{11}$$

### 4.2 Poseidon2: Polynomial Modeling

POSEIDON and Poseidon2 differ in the linear layer, constant addition layer for partial rounds, and the initial round. The first round can be modeled as follows:

$$x_{1,j} - \left( \sum_{k=1}^{t} \mathcal{M}''[j,k] \cdot x_{0,k} + c_{1,j} \right)^{\alpha} = 0 \qquad j \in [1,r],$$

The full rounds are modeled in the same way as Section 4.1 with different coefficients coming from $\mathcal{M}''$. The partial rounds are modeled as follows:

$$\begin{aligned} x_{i,1} - (y_1 + c_{i,1})^{\alpha} &= 0 \\ y_j = \mathcal{M}'_{j,1} \cdot x_{i,1} + \sum_{k=2}^{t} \mathcal{M}'_{j,k} \cdot (y_k) & \qquad j \in [1,t]. \end{aligned}$$

Where **Y** is defined in the same way as Section 4.1. The final system, similar to POSEIDON's system, has $r + (R_F - 1)t + R_P$ polynomials of degree $\alpha$ in $r + (R_F - 1)t + R_P$ variables. The solving degree of Poseidon2 is equal to POSEIDON when $R_F > 2$ and hence Equation (11) is the linear regression of the collected data for solving degree.

## 4.3 Complexity of the Attack and Broken Parameters

The complexity of computing Gröbner basis for different instances of POSEIDON is determined by the solving degrees. Our experiments confirm that POSEIDON and Poseidon2 can be modeled as a sequence of polynomials that are not regular. Therefore, the Macaulay bound used in [29] overestimates the resistance against the Gröbner basis attacks.

We show that partial rounds do not provide the same level of resistance against algebraic attacks as full rounds. More specifically, the complexity of the attack is a function of the number of S-boxes rather than the number of rounds and the partial rounds increase the solving degree by at most one in each round. Moreoverm There are cases where partial rounds do not increase the solving degree at all.

We calculated the solving degree for more than 100 different parameters of POSEIDON and Poseidon2, with state size up to 5 and rate up to 4. Our experiments show that the degree of the non-linear transfer, $\alpha$, does **not** affect the *growth* of the solving degree and only affects the constant number in the affine equation of the upper bound. The field size also has no effect on the solving degree based on our experiments. In the case of Poseidon2, the solving degree is the same as POSEIDON's solving degree when $R_F > 2$. Using the collected data, and the fact that the solving degree grows linearly as a function of number of variables, we extrapolate the solving degree as follows:

$$d_{sol} = r\frac{R_F}{2} + R_P + \alpha. \tag{12}$$

In Figure 4, we compared the solving degree, our upper bound for solving degree, and the Macaulay bound (POSEIDON's claimed degree) in case of growth in the number of partial rounds.

Using $d_{sol}$, the complexity of the Gröbner basis attack can be summarized as:

1. **Computing the Gröbner Basis in *degrevlex* Order.**

$$\binom{(R_F-1)t + R_P + r + r\frac{R_F}{2} + R_P + \alpha}{r\frac{R_F}{2} + R_P + \alpha}^{2.3727}$$

2. **Changing the Term Order to *lex* Order.** The degree of the corresponding zero-dimensional ideal is:

$$d_I = \alpha^{r \cdot R_F + R_P},$$

and the asymptotic complexity of the sparse FGLM is:

$$O\left( \sqrt{\frac{6}{((R_F-1)t + R_P + r)\pi}} (\alpha)^{\left(2 + \frac{(R_F-1)t+R_P+r-1}{(R_F-1)t+R_P+r}\right) \cdot d_I} \right).$$

3. **Complexity of Finding the Variety.** The largest degree of polynomials in the final Gröbner basis with respect to *lex* order is:

$$d_{\text{GB}} = \alpha^{r \cdot (R_F - 1) + R_P},$$

and the complexity of finding the variety is:

$$O\left( (d_{\text{GB}})^2 \left(\log(d_{\text{GB}}) \log\log(d_{\text{GB}})\right) \left(\log p + \log(d_{\text{GB}})\right) \right).$$

In Table 3, we describe instances of POSEIDON and Poseidon2 that the complexity of Gröbner basis attack is less than their claimed security level.

| $\lambda$ | $\alpha$ | $t$ | $r$ | $R_F$ | $R_P$ | $\mathcal{C}_{\text{GB}}$ | $\mathcal{C}_{\text{SFGLM}}$ | $\mathcal{C}_{\text{Elim}}$ |
|---|---|---|---|---|---|---|---|---|
| 1024 | 3 | 24 | 8 | 8 | 85 | **731.77** | **705.67** | **466.18** |
| 512 | 5 | 12 | 4 | 8 | 57 | **435.10** | 615.40 | 413.62 |
| 384 | 7 | 9 | 3 | 8 | 47 | **361.81** | 593.25 | 400.61 |

Table 3: Examples of POSEIDON and Poseidon2 hash functions with security parameter $\lambda$ over the finite field $\mathbb{F}_p$ with $\log_2(p) \approx 128$. $\mathcal{C}_{\text{GB}}$ is the complexity of computing the Gröbner basis in *degrevlex* order, $\mathcal{C}_{\text{SFGLM}}$ is the asymptotic complexity of sparse FGLM, and $\mathcal{C}_{\text{Elim}}$ is the complexity factoring univariate polynomials and recovering their roots.

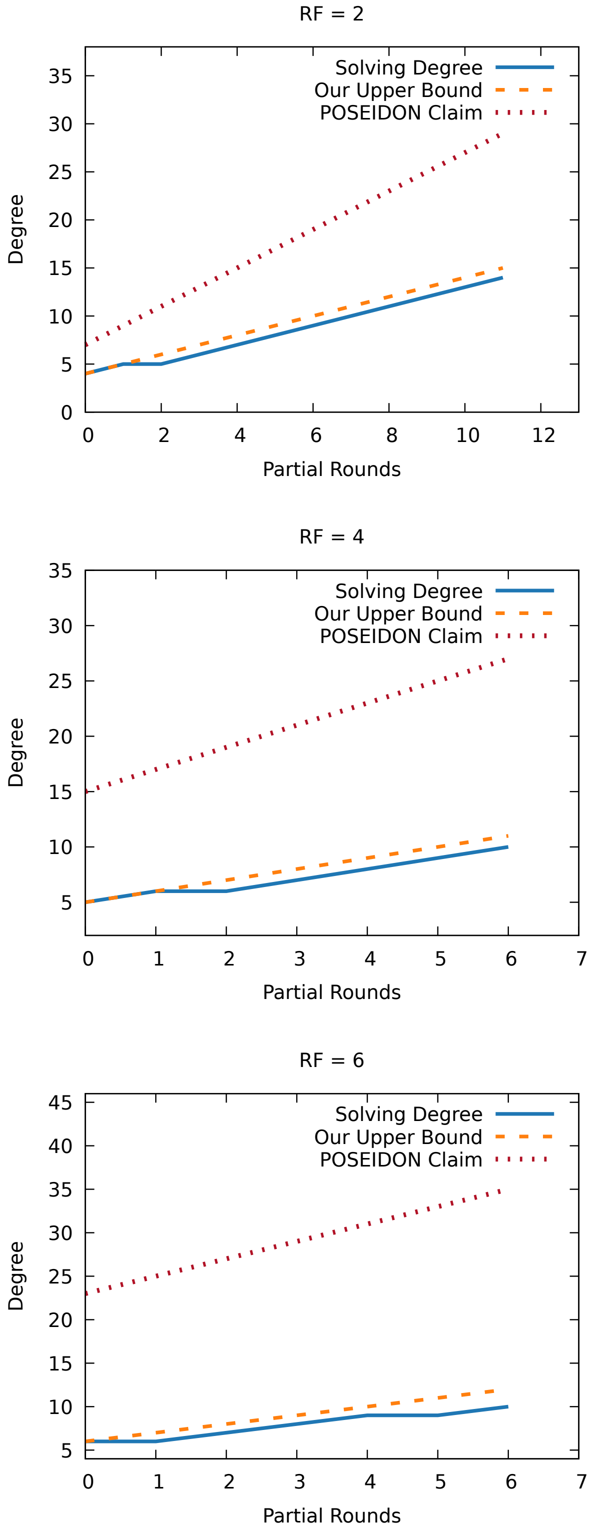

Figure 4: The comparison of our upper bound (dashed line), the solving degree, and the Macaulay bound used to argue POSEIDON security. In all three plots, $t = 2, p = 65519$, and $R_F$ is fixed.

## 5 The XL Attack

Let $\mathcal{P}$ denote the polynomial system describing POSEIDON. Then for the parameter sets $(\alpha, t, r, R_F, R_P)$, $\mathcal{P}$ is a system with $N = r + (R_F - 1)t + R_P$ polynomials of degree $\alpha$ in $N = r + (R_F - 1)t + R_P$ variables. To perform the XL attack, we compute the smallest degree $D$ such that extending $\mathcal{P}$ results in an over-determined Macaulay matrix. More precisely, the degree $D$ is the smallest non-negative integer such that:

$$N\binom{N+D-\alpha}{D-1} \geq \binom{N+D}{D-1}.$$

The degree $D$ ensures that the Macaulay matrix of the extended system is over-determined and can be solved using linear algebra operations. In Table 4, some instances that are vulnerable to XL attack is described. These results apply to both Poseidon and Poseidon2.

| $\lambda$ | $\log_2(p)$ | $\alpha$ | $t$ | $r$ | $R_F$ | $R_P$ | $D$ | $\mathcal{C}_{\text{XL}}$ |
|---|---|---|---|---|---|---|---|---|
| 1024 | 128 | 3 | 24 | 8 | 8 | 85 | 50 | **459.87** |
| 512 | 64 | 3 | 24 | 8 | 8 | 42 | 45 | **402.64** |

Table 4: Instances of POSEIDON and Poseidon2 hash functions with security parameter $\lambda$ over the finite field $\mathbb{F}_p$. $\mathcal{C}_{\text{XL}}$ is the upper bound of the complexity of the XL attack and $D$ is the degree of the extended system.

## 6 Discussion

We conclude this work by summarizing the results, the steps taken toward disclosure and possible future research directions.

### 6.1 Disclosure

The results presented in section 3.4 and section 4 were shared with the designers. They have confirmed them and updated their documentation to provide concrete information about which instances of POSEIDON and Poseidon2 are safe to use. They were subsequently uploaded to ePrint and shared on Twitter by one of the designers. Due to time constraints, the XL attack presented in section 5 was not shared with the designers and this will be done in parallel to the review process.

Since the vulnerabilities presented in this paper are more pronounced in non-standard security levels we are not aware of any practical instance directly affected and therefore do not call for immediate action. However, we encourage the potential users to take into account in their risk assessment the ongoing erosion in the security of HADES instances.

### 6.2 Future Research Directions

In the pursuit of designing more secure ciphers, further investigation into the behavior of symmetric primitives with regard to algebraic attacks, as well as the analysis of the interplay between theoretical complexities and the actual running time of such attacks, is highly valuable. Within the scope of our work, we employed conservative upper bounds for determining the complexity of the attacks. In the case of the Gröbner basis attacks, we chose to neglect instances in which the solving degree remains static during some rounds. Hence, we posit that by adopting a less restrictive approach, we may improve the efficacy of the attack and be able to successfully break a larger set of parameters. Another possible direction of research is to investigate the resistance of partial SPN layers against algebraic attacks and quantify the complexity of algebraic attacks as a function of the number of rounds and the number of S-boxes in order to design efficient and secure primitives. Finally, no attempt was made to optimize the second and third steps of the Gröbner basis attack despite these steps sometimes being the bottleneck for a successful attack (see table 3).

### 6.3 Summary

In this paper, we analyzed the security of POSEIDON and Poseidon2 which are primitives based on the HADES design strategy. We studied two categories of algebraic attacks, namely XL attacks and Gröbner basis attacks, and showed that partial rounds are not providing the claimed resistance. Using Gröbner basis attacks, we break instances of POSEIDON and Poseidon2 claiming 1024 bits of security using an attack whose complexity is upper bounded by $2^{731.77}$ and show that the original security argument does not hold for instances with as small as 384 bits of claimed security. Using the XL attack, we break instances of POSEIDON and Poseidon2 with 512 bits of security using an attack whose complexity is upper bounded by $2^{402.64}$.

## References

[1] Martin R. Albrecht, Carlos Cid, Lorenzo Grassi, Dmitry Khovratovich, Reinhard Lüftenegger, Christian Rechberger, and Markus Schofnegger. Algebraic cryptanalysis of stark-friendly designs: Application to marvellous and mimc. Cryptology ePrint Archive, Paper 2019/419, 2019. https://eprint.iacr.org/2019/419.

[2] Martin R. Albrecht, Lorenzo Grassi, Christian Rechberger, Arnab Roy, and Tyge Tiessen. MiMC: Efficient Encryption and Cryptographic Hashing with Minimal Multiplicative Complexity. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part I*, volume 10031 of *Lecture Notes in Computer Science*, pages 191–219, 2016.

[3] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. Ciphers for mpc and fhe. Cryptology ePrint Archive, Paper 2016/687, 2016. https://eprint.iacr.org/2016/687.

[4] Abdelrahaman Aly, Tomer Ashur, Eli Ben-Sasson, Siemen Dhooghe, and Alan Szepieniec. Design of Symmetric-Key Primitives for Advanced Cryptographic Protocols. *IACR Trans. Symmetric Cryptol.*, 2020(3):1–45, 2020.

[5] Tomer Ashur, Al Kindi, Willi Meier, Alan Szepieniec, and Bobbin Threadbare. Rescue-prime optimized. Cryptology ePrint Archive, Paper 2022/1577, 2022. https://eprint.iacr.org/2022/1577.

[6] Tomer Ashur, Mohammad Mahzoun, and Dilara Toprakhisar. Chaghri — an fhe-friendly block cipher. Cryptology ePrint Archive, Paper 2022/592, 2022. https://eprint.iacr.org/2022/592.

[7] JP Aumasson, Dmitry Khovratovich, Bart Mennink, and Porçu Quine. Safe: Sponge api for field elements. Cryptology ePrint Archive, Paper 2023/522, 2023. https://eprint.iacr.org/2023/522.

[8] Magali Bardet, Jean-Charles Faugère, and Bruno Salvy. On the complexity of the f5 gröbner basis algorithm, 2013.

[9] Augustin Bariant, Clémence Bouvier, Gaëtan Leurent, and Léo Perrin. Algebraic attacks against some arithmetization-oriented primitives. *IACR Transactions on Symmetric Cryptology*, 2022(3):73–101, Sep. 2022.

[10] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. On the indifferentiability of the sponge construction. In Nigel Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, pages 181–197, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[11] Luk Bettale, Jean-Charles Faugère, and Ludovic Perret. Solving polynomial systems over finite fields: improved analysis of the hybrid approach. In Joris van der Hoeven and Mark van Hoeij, editors,

*International Symposium on Symbolic and Algebraic Computation, ISSAC'12, Grenoble, France - July 22 - 25, 2012*, pages 67–74. ACM, 2012.

[12] Tim Beyne, Anne Canteaut, Itai Dinur, Maria Eichlseder, Gregor Leander, Gaëtan Leurent, María Naya-Plasencia, Léo Perrin, Yu Sasaki, Yosuke Todo, and Friedrich Wiemer. Out of oddity – new cryptanalytic techniques against symmetric primitives optimized for integrity proof systems. Cryptology ePrint Archive, Paper 2020/188, 2020. https://eprint.iacr.org/2020/188.

[13] Christina Boura, Anne Canteaut, and Christophe De Cannière. Higher-order differential properties of keccak and luffa. Cryptology ePrint Archive, Paper 2010/589, 2010. https://eprint.iacr.org/2010/589.

[14] Clémence Bouvier, Pierre Briaud, Pyrros Chaidos, Léo Perrin, Robin Salen, Vesselin Velichkov, and Danny Willems. New design techniques for efficient arithmetization-oriented hash functions:anemoi permutations and jive compression mode. Cryptology ePrint Archive, Paper 2022/840, 2022. https://eprint.iacr.org/2022/840.

[15] Alessio Caminata and Elisa Gorla. Solving degree, last fall degree, and related invariants. *CoRR*, abs/2112.05579, 2021.

[16] Anne Canteaut, Sergiu Carpov, Caroline Fontaine, Tancrède Lepoint, María Naya-Plasencia, Pascal Paillier, and Renaud Sirdey. Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression. Cryptology ePrint Archive, Paper 2015/113, 2015. https://eprint.iacr.org/2015/113.

[17] David G. Cantor and Hans Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, 36(154):587–592, 1981.

[18] Carlos Cid, John Petter Indrøy, and Håvard Raddum. Fasta - a stream cipher for fast fhe evaluation. Cryptology ePrint Archive, Paper 2021/1205, 2021. https://eprint.iacr.org/2021/1205.

[19] Orel Cosseron, Clément Hoffmann, Pierrick Méaux, and François-Xavier Standaert. Towards globally optimized hybrid homomorphic encryption - featuring the elisabeth stream cipher. Cryptology ePrint Archive, Paper 2022/180, 2022. https://eprint.iacr.org/2022/180.

[20] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In Bart Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, pages 392–407, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[21] David Cox, John Little, and Donal O'Shea. Ideals, varieties, and algorithms. an introduction to computational algebraic geometry and commutative algebra. 2007.

[22] Jintai Ding and Dieter Schmidt. *Solving Degree and Degree of Regularity for Polynomial Systems over a Finite Fields*, pages 34–49. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[23] Christoph Dobraunig, Maria Eichlseder, Lorenzo Grassi, Virginie Lallemand, Gregor Leander, Eik List, Florian Mendel, and Christian Rechberger. Rasta: A cipher with low anddepth and few ands per bit. Cryptology ePrint Archive, Paper 2018/181, 2018. https://eprint.iacr.org/2018/181.

[24] Christoph Dobraunig, Lorenzo Grassi, Lukas Helminger, Christian Rechberger, Markus Schofnegger, and Roman Walch. Pasta: A case for hybrid homomorphic encryption. Cryptology ePrint Archive, Paper 2021/731, 2021. https://eprint.iacr.org/2021/731.

[25] J.C. Faugère, P. Gianni, D. Lazard, and T. Mora. Efficient computation of zero-dimensional gröbner bases by change of ordering. *Journal of Symbolic Computation*, 16(4):329–344, 1993.

[26] Jean-Charles Faugère and Chenqi Mou. Sparse fglm algorithms. *Journal of Symbolic Computation*, 80:538–569, 2017.

[27] B. Gérard, Vincent Grosso, M. Naya-Plasencia, and François-Xavier Standaert. Block ciphers that are easier to mask: How far can we go? In Guido Bertoni and Jean-Sébastien Coron, editors, *Cryptographic Hardware and Embedded Systems - CHES 2013*, pages 383–399, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[28] Lorenzo Grassi, Yonglin Hao, Christian Rechberger, Markus Schofnegger, Roman Walch, and Qingju Wang. Horst meets fluid-spn: Griffin for zero-knowledge applications. Cryptology ePrint Archive, Paper 2022/403, 2022. https://eprint.iacr.org/2022/403.

[29] Lorenzo Grassi, Daniel Kales, Dmitry Khovratovich, Arnab Roy, Christian Rechberger, and Markus Schofnegger. Starkad and poseidon: New hash

functions for zero knowledge proof systems. *IACR Cryptol. ePrint Arch.*, page 458, 2019.

[30] Lorenzo Grassi, Dmitry Khovratovich, and Markus Schofnegger. Poseidon2: A faster version of the poseidon hash function. Cryptology ePrint Archive, Paper 2023/323, 2023. https://eprint.iacr.org/2023/323.

[31] Lorenzo Grassi, Reinhard Lüftenegger, Christian Rechberger, Dragos Rotaru, and Markus Schofnegger. On a generalization of substitution-permutation networks: The HADES design strategy. In Anne Canteaut and Yuval Ishai, editors, *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part II*, volume 12106 of *Lecture Notes in Computer Science*, pages 674–704. Springer, 2020.

[32] Jincheol Ha, Seongkwang Kim, Byeonghak Lee, Jooyoung Lee, and Mincheol Son. Rubato: Noisy ciphers for approximate homomorphic encryption (full version). Cryptology ePrint Archive, Paper 2022/537, 2022. https://eprint.iacr.org/2022/537.

[33] Phil Hebborn and Gregor Leander. Dasta - alternative linear layer for rasta. *IACR Trans. Symmetric Cryptol.*, 2020(3):46–86, 2020.

[34] Thomas Jakobsen and Lars R. Knudsen. The interpolation attack on block ciphers. In Eli Biham, editor, *Fast Software Encryption*, pages 28–40, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.

[35] Nathan Keller and Asaf Rosemarin. Mind the middle layer: The hades design strategy revisited. Cryptology ePrint Archive, Paper 2020/179, 2020. https://eprint.iacr.org/2020/179.

[36] Lars R. Knudsen. Truncated and higher order differentials. In Bart Preneel, editor, *Fast Software Encryption*, pages 196–211, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg.

[37] F. S. MacAulay. Some formulæ in elimination. *Proceedings of the London Mathematical Society*, s1-35(1):3–27, 1902.

[38] F. S. MacAulay. Some formulæ in elimination. *Proceedings of the London Mathematical Society*, s1-35(1):3–27, 1902.

[39] Pierrick Méaux, Anthony Journault, François-Xavier Standaert, and Claude Carlet. Towards stream ciphers for efficient fhe with low-noise ciphertexts. Cryptology ePrint Archive, Paper 2016/254, 2016. https://eprint.iacr.org/2016/254.

[40] Jan Ferdinand Sauer. Gröbner basis-attacking a tiny sponge. Technical report, AS Discrete Mathematics, 2021. https://asdm.gmbh/2021/06/28/gb_experiment_summary/.

[41] Victor Shoup. Factoring polynomials over finite fields: Asymptotic complexity vs. reality. 1993.

[42] Alan Szepieniec, Tomer Ashur, and Siemen Dhooghe. Rescue-prime: a standard specification (sok). *IACR Cryptol. ePrint Arch.*, page 1143, 2020.

[43] Alan Szepieniec, Alexander Lemmens, Jan Ferdinand Sauer, Bobbin Threadbare, and Al-Kindi. The tip5 hash function for recursive starks. Cryptology ePrint Archive, Paper 2023/107, 2023. https://eprint.iacr.org/2023/107.

[44] Virginia Williams. Breaking the coppersmith-winograd barrier. 09 2014.

[45] A.M. Youssef, S. Mister, and Stafford Tavares. On the design of linear transformations for substitution permutation encryption networks. pages 40–48, 09 1997.