Starkad and Poseidon: New Hash Functions for Zero Knowledge Proof Systems

Lorenzo Grassi[1], Daniel Kales[1], Dmitry Khovratovich[2,3], Arnab Roy[4], Christian Rechberger[1], and Markus Schofnegger[1]

[1] IAIK, Graz University of Technology
[2] Evernym Inc.
[3] ABDK Consulting
[4] University of Bristol, Bristol, UK
firstname.lastname@iaik.tugraz.at, khovratovich@gmail.com, arnab.roy@bristol.ac.uk

Abstract. The area of practical proof systems, like SNARKs, STARKs, or Bulletproofs, is seeing a very dynamic development. Many use-cases of such systems involve, often as their most expensive apart, proving the knowledge of a preimage under a certain cryptographic hash function.

In this paper we present a modular framework and concrete instances of cryptographic hash functions which either work natively with GF(p) objects or on binary strings. Compared to competitors, our hash function Poseidon uses up to 8x fewer constraints per message bit compared to Pedersen Hash, whereas our STARK-friendly hash Starkad takes wins the factor of 4 over the hash function Friday by using a much smaller field.

Table of Contents

1 Introduction

The recent advances in computational integrity proof systems made a number of computational tasks verifiable in short time and/or in zero knowledge. Several protocols appeared that require one party to prove the knowledge of a seed-derived secret, of an element being part of a large set, or their combination. Whereas accumulator-based solutions [13,12] and algebraic Schnorr proofs exist in the area, they are quite involving and thus error-prone, require a trusted setup, are limited in statement language, and often are slow. An alternative is to express secret derivation via application of cryptographic hash functions, and to prove set membership via presenting an opening in a properly chosen Merkle tree, also built on a cryptographic hash function. Such hash-based protocols require a computational integrity proof system, which can be applied to an arbitrary arithmetic circuit. However, for the protocol to be efficient, proofs must be generated and verified in reasonable time, which in turn require the hash function to be cheap in certain (proofsystem-dependent) metric.

At the beginning of 2019, the most popular proof systems are ZK-SNARKs [35,20], Bulletproofs [11], and ZK-STARKs [6]. The former two have been already applied to a number of real-world protocols, whereas the latter is the most promising from the perspective of performance and post-quantum security. These three systems use two quite different metrics:

- R1CS metric is used in ZK-SNARKs and Bulletproofs. The computation i s expressed as an arithmetic circuit with multiplication and addition gates over a prime field $GF(p)$. The circuit is then converted to a system of rank-1 constraints of form $L_1(X) \cdot L_2(X) = L_3(X)$, where X is the tuple of internal and input variables, L_i are linear forms and $\cdot$ is the field multiplication. The proof generation complexity is directly proportional to the number T of constraints, which often corresponds to the number of multiplicative gates. The prime field $GF(p)$ is the scalar field of an elliptic curve, where for ZK-SNARKs the curve should be pairing friendly and for Bulletproofs it should be just a secure curve.
- AET metric is used in ZK-STARKs. The computation is expressed as a set of internal program states related to each other by polynomial equations of degree d. The state consists of w binary $GF(2^n)$ field elements and undergoes T transformations. The proof generation is roughly proportional to the product $w \cdot d \cdot T$, whereas n should be 32 or higher. The number and sparsity of polynomial constraints do not play a major role.

Our goal was to design a family of hash functions that are optimal in either R1CS or AET metric, for different p and $GF(2^n)$ sizes. Even though the metrics are different we tried to make the hash functions to share as many components as possible to reuse the analysis. It turned out that the Substitution-Permutation-Network (SPN) design, well known in symmetric cryptography, allows a generic hash function framework where the only security-critical parameter that has to be changed for each instance is the number of rounds, but we provide an efficient and transparent strategy for its choice. The S-Box is almost universally chosen as the cube function x^3. The only exception is the case of a field where this function is not a bijection, for which we suggest another S-Box, $1/x$.

Our contributions. We design and analyze two families of hash functions: Starkad and Poseidon, which are both based on the HadesMiMC strategy [32]. The latter is a permutation design with t field elements forming the internal state, and each round is a composition of the S-Box layer, linear transformation, and a round constant addition. We aim to support 128- and 256-bit security, where the security level is the same for collision and preimage resistance. For each pair (basic field, security level) we suggest a concrete instance of either Starkad (for binary field) or Poseidon (for prime field) permutation. Some middle rounds (called *partial*) carry only 1 rather than t S-Boxes to save up R1CS or AET cost. Each hash function is a certain permutation in the sponge mode of operation, where a few Sbox elements are reserved for the capacity (roughly double the security level in bits) and the rest for the rate. The permutation width is set close to 1500 bits so the rate/capacity ratio is 2 for 256-bit security level and 5 for 128-bit security level. The Merkle trees built on these hash functions will thus have branching also 2 or 5, respectively.

We provide an extensive cryptanalysis of both families with an accent on algebraic methods as these prove to be the most effective. We explore different variants of interpolation, Grobner basis, higher-order differential attacks. As our permutations are quite wide, we do not aim that they would behave as randomly chosen permutations. Instead, for security level of M bits we require that no attack could exhibit a non-random property of a permutation faster than in 2^M queries. We then calculate the maximum number of rounds for each field, security level, and fixed permutation width that can be attacked. Then we select the number of rounds for concrete instances using a percentage security margin.

We have evaluated the number of constraints in Poseidon instances for the R1CS metric and the STARK complexity in Starkad instances for the AET metric. Our primary proposals Poseidon-252, Poseidon-256, and Starkad-252 are listed in Table 1 and are compared to similar-purpose designs.

Related work. The Zcash designers introduced a new 256-bit hash function called Pedersen hash [22, p.134], which is effectively a vectorized Pedersen commitment in elliptic curve groups with short vector elements. For the claimed 128-bit security level, it utilizes 869 constraints per 516-bit message chunks, thus having 1.7 constraints per bit, whereas our Poseidon instances use from 0.2 to 0.45 constraints per bit, depending on the underlying prime field.

For the binary field case, Ashur and Dhooghe [3] have recently introduced the STARK-friendly blockcipher Jarvis and its derivative hash function Friday with several instances, from 128 to 256-bit security levels. They use an SPN structure with a single inverse S-Box, followed by an affine transformation (with low degree in the extension field).

Structure of the paper. We provide an overview of our design strategy in Section 2. We summarize the cryptanalysis results in Section 3 with the details in Appendix. We explain the rationale for the choice of the number of rounds in Section 4. Then we suggest concrete parameters (permutation size, number of rounds, round constant generation) for our designs Starkad and Poseidon in Section 5. We estimate R1CS costs of Starkad instances in Section 6 and AET (STARK) costs in Section 7.

2 Starkad and Poseidon Hash Functions

2.1 Overview

In the following we propose two hash functions:

- the hash function[5] Starkad-Hash for the *binary case* is constructed by instantiating a sponge construction [7] with Starkad-Permutation – denoted by Starkad$^\pi$;
- the hash function[6] Poseidon-Hash for the *prime case* is constructed by instantiating a sponge construction [7] with Poseidon-Permutation – denoted by Poseidon$^\pi$.

Both permutations are variants of HadesMiMC, a block cipher proposed in [32] instantiated by a fixed key, e.g. 0^κ.

We recall that when the internal permutation $\mathcal{P}$ of an N-bit sponge function (composed of c-bit capacity and r-bit bitrate: $N = c + r$) is modeled as a randomly chosen permutation, it has been proven by Bertoni *et al.* [7] to be indifferentiable from a random oracle up to $2^{c/2}$ calls to $\mathcal{P}$. In other words, a sponge with a capacity of c provides $2^{c/2}$ collision and $2^{c/2}$ (second) preimage resistance. Given a permutation of size N and a desired security level s, we can hash $r = N - 2s$ bits per call to the permutation. Following this design strategy, we choose the number of rounds of the inner permutations Poseidon$^\pi$ and Starkad$^\pi$ in order to ensure that such permutation does *not* exhibits non-generic property up to 2^M queries[7], where M is the desired security level.

As usual, the message is first padded according to the sponge specification so that the number of message blocks is a multiple of r, where r is the rate in the sponge mode. In our case, we use Poseidon$^\pi$ or Starkad$^\pi$ permutation, where $N \geq 4 \cdot M$ (M is the security level). For Poseidon-256 (analogous for

[5] *About the name:* Starkad was a legendary hero in Norse mythology, who used to hash his enemies with 2^2 swords in 2^3 arms.

[6] *About the name:* Poseidon – brother of Zeus and Hades – was god of the Sea and other waters, of earthquakes and of horses.

[7] In other words, such permutation can not be distinguished from a randomly-drawn permutation.

Table 1: Our primary proposals and their competitors. The R1CS/bit and AET/bit costs are obtained by dividing the R1CS (resp. AET) prover costs by message rate. Note that AET costs are measured in field operations, whose costs in software/hardware grow quadratically with field size.

Name	**S-box**	**Security** M	**Rate** $n \cdot t - 2M$	**Prime** $(\log_2 p)$	**# SB** (t)	$\mathbf{R_F}$	$\mathbf{R_P}$	**Field**	**R1CS** /perm.	**R1CS** /bit
Poseidon-252	x^{-1}	126	1260	252	6	8	127	Ristretto	525	0.42
Poseidon-256	x^3	128	1280	256	6	8	84	$p = 2 \pmod 3$	264	0.21
Pedersen Hash	-	128	516	-	-	-		BLS12-381	869	1.68
Name	**S-box**	**Security** M	**Rate** $n \cdot t - 2M$	**SB size**	**# SB** (t)	$\mathbf{R_F}$	$\mathbf{R_P}$	**Field**	**AET** /perm.	**AET** /bit
Starkad-252	x^3	126	1260	63	24	12	39	$GF(2^{63})$	67676	54
Friday-256	x^{-1}	128	256	-	-	-		$GF(2^{256})$	10371	41

Starkad-256), we thus use Poseidon permutation with $N = n \cdot t \geq 1024$. The capacity is chosen as 512. This choice allows e.g. for processing more input bits then SHA-256 (512 bits) while at the same time offering collision security and (second) preimage security of 256 bits. Similar considerations hold as well for Poseidon-128 and/or Starkad-128.

2.2 The Hades Strategy

(Cryptographic) Permutations are typically designed by iterating an efficiently implementable round function many times in the hope that the resulting composition behaves like a randomly drawn permutation. In general, *the same round function is iterated enough times to make sure that any symmetries and structural properties that might exist in the round function vanish.*

Instead of considering the same round function in order to construct the cipher (to be more precise, the same non-linear layer for all rounds), in [32] authors propose to consider *a variable number of S-Boxes per round*, that is, to use different S-Box layers in the round functions.

Similar to any other SPN design, each round of a cipher based on Hades is composed of three steps:

1. *Add-Round Key* - denoted by $ARK(\cdot)$;
2. *SubWords* operation - denoted by S-Box$(\cdot)$;
3. *MixLayer* - denoted by $M(\cdot)$.

A final round key addition is usually added after the last round, but we do not use this in the Starkad/Poseidon hash functions for uniformity:

$$\underbrace{ARK \to \text{ S-Box} \to M}_{1st \text{ round}} \to \ldots \to \underbrace{ARK \to \text{ S-Box} \to M}_{(R-1)\text{-}th \text{ round}} \to \underbrace{ARK \to \text{ S-Box} \to M}_{R\text{-}th \text{ round}}$$

The crucial property of HADES is that *the number of S-Boxes per round is not the same for every round*:

- a certain number of rounds - denoted by R_F - has a *full* S-Box layer, i.e., t S-Box functions;
- a certain number of rounds - denoted by R_P - has a *partial* S-Box layer, i.e., $1 \le s < t$ S-Boxes and $(t-s)$ identity functions.

In the following, we limit to consider only the case $s = 1$, that is, R_P rounds have a single S-Box per round and $t-1$ identity functions.

In more details, assume $R_F = 2 \cdot R_f$ is an even number[8]. Then

- the first R_f rounds have a full S-Box layer,
- the middle R_P rounds have a partial S-Box layer (i.e., 1 S-Box layer),
- the last R_f rounds have a full S-Box layer.

Figure 1 shows the strategy HADES. Note that the rounds with a partial S-Box layer are "masked" by the rounds with a full S-Box layer, which means that an attacker should not (directly) take advantage of the rounds with a partial S-Box layer.

Behind HADES Strategy. The crucial point of our design is that it contains *both rounds with full S-Box layers and rounds with partial S-Box layers*. This allows to provide *simpler argumentation about the security against statistical attacks* than the one proposed for P-SPN ciphers.

In more details, a certain number of rounds $R_F^{stat} = 2 \cdot R_f^{stat}$ with full S-Box layer situated at the beginning and the end guarantee security against statistical attacks. Indeed, even without the middle part, they are sufficient in order to apply the "Wide-Trail" strategy, in a way that we are going to show in the following. Security against all algebraic attacks is achieved working both with rounds $R_F = R_F^{stat} + R'_F \ge R_F^{stat}$ with full S-Box layer and rounds $R_P \ge 0$ with partial S-Box layer. Even if few (even one) S-Boxes per round are potentially sufficient to increase the degree of the encryption/decryption function (which mainly influences the cost of an algebraic attack), other factors can play a crucial role on the cost of such attacks (e.g. a Gröbner basis attack depends also on the number of non-linear equation to solve).

With this in mind, the idea is to construct "something in the middle" between an SPN and a P-SPN cipher. Moreover, since we aim to have the same security w.r.t. chosen-plaintext and chosen-ciphertext attacks, we consider a cipher which is "symmetric": in other words, the same number of rounds with full non-linear

[8] $R_F = 2 \cdot R_f$ is even in order to have a "symmetric" permutation. Note that some attacks – like the statistical ones – have the same performance both in the forward and in the backward direction. Thus a "symmetric" permutation with $R_F = 2 \cdot R_f$ guarantees the same security against these attacks both in the chosen-/known-"plaintext" scenario and in the chosen-/known-"ciphertext" one.

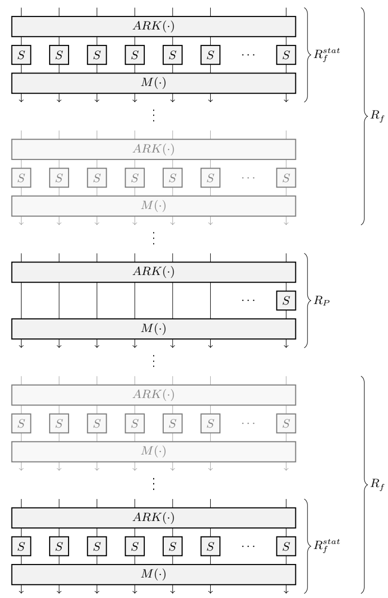

Fig. 1: Construction of Hades.

layer are applied at the beginning and at the end, where the rounds with partial non-linear layers are in the middle and they are “masked” by the rounds with full non-linear layers. As a result, depending on the cost metric that one aims to minimize (e.g. the total number of non-linear operations) and on the size of the S-Box, in the following we provide the *best ratio* between the number of rounds with full S-Box layer and with partial ones in order to both achieve security and minimize the cost metric.

For more details about Hades strategy, we refer to [32].

What about the choice of the linear and of the non-linear layer? This strategy does not pose any restriction/constriction on the choice of the linear layer and/or on the choice of the S-Box. The idea is to *consider a “traditional” SPN cipher based on the wide trail strategy, and then to replace a certain number of rounds with full S-Box layer with the same number of rounds with partial S-Box layer* in order to minimize the number of non-linear operations, but without affecting the security. The Hades strategy has a huge impact especially in the case of ciphers with low-degree S-Box, since in this case a large number of rounds is required to guarantee security against algebraic attacks.

2.3 Permutations Starkad$^\pi$ & Poseidon$^\pi$

HadesMiMC is a block cipher constructed using the strategy just proposed, hence it is both an SPN and a Partial-SPN cipher. Roughly speaking, HadesMiMC is obtained by applying the HADES strategy to the cipher SHARK [36], proposed by Rijmen *et al.* in 1996 and based on the wide trail strategy.

HADESMiMC works with texts of $t \geq 2$ words[9] in $\mathbb{F}_p$ or $\mathbb{F}_{2^n}$, where p is a prime of size $p \approx 2^n$.

As for SHARK, the MixLayer of HADESMiMC is simply defined by a multiplication with a fixed $t \times t$ MDS matrix or near-MDS matrix. The number of rounds $R = 2 \cdot R_f + R_P$ depends on the choice of the S-Box and of the parameters n and t. For the applications that we have in mind, we focus on

- the cubic one: S-Box$(x) = x^3$ – remember that the cubic S-Box is a bijection in $GF(2^n)$ iff n is odd and it is a bijection in $GF(p)$ iff $p = 2 \mod 3$; in the following, we call this case as HADESCUBIC, whereas the permutations are called $x^3 - \text{POSEIDON}^\pi$ for the prime case and $x^3 - \text{STARKAD}^\pi$ for the binary case;
- the quintic one S-Box$(x) = x^5$ – remember that the x^5 S-Box is a bijection in $GF(2^n)$ iff $2^n \neq 1 \mod 5$, and it is a bijection in $GF(p)$ iff $p \neq 1 \mod 5$; in the following, we call this case as HADESFIFTH, whereas the permutations are called $x^5 - \text{POSEIDON}^\pi$ for the prime case and $x^5 - \text{STARKAD}^\pi$ for the binary case;
- the inverse one S-Box$(x) = x^{-1}$; in the following, we call this case as HADESINVERSE, whereas the permutations are called $x^{-1} - \text{POSEIDON}^\pi$ for the prime case and $x^{-1} - \text{STARKAD}^\pi$ for the binary case.

About the MDS Matrix. A $t \times t$ MDS matrix[10] with elements in $GF(2^n)$ (or $GF(p)$ where $p \approx 2^n$) exists if the condition (see [29] for details)

$$\log_2(2t + 1) \leq n$$

(or equivalently $t \cdot \log_2(2t + 1) \leq N$) is satisfied.

Given n and t, there are several ways to construct an MDS matrix. One of them is using Cauchy Matrix [37], which we recall here briefly. Let $x_i, y_i \in \mathbb{F}_{2^n}$ for $i = 1, ..., t$ s.t.

- $\forall i \neq j: \qquad x_i \neq x_j, \quad y_i \neq y_j,$
- for $1 \leq i \leq t$ and $1 \leq j \leq t$: $\qquad x_i \oplus y_j \neq 0.$

To fulfill these conditions, one can simply consider x_i s.t. the $t - \log_2(t)$ most significant bits are zero. Then, choosing $r \in \mathbb{F}_{2^n}$ s.t. the $t - \log_2(t)$ most significant

[9] The case $t = 1$ corresponds to MiMC [2].

[10] A matrix $M \in \mathbb{F}^{t \times t}$ is called *Maximum Distance Separable* (MDS) matrix iff it has branch number $\mathcal{B}(M)$ equal to $\mathcal{B}(M) = t + 1$. The branch number of M is defined as $\mathcal{B}(M) = \min_{x \in \mathbb{F}^t} \{wt(x) + wt(M(x))\}$, where wt is the hamming weight. Equivalently, a matrix M is MDS iff every submatrix of M is non-singular.

bits are non zero, let $y_i = x_i \oplus r$. Let A be the Cauchy matrix defined by

$$a_{i,j} = \frac{1}{x_i \oplus y_j}.$$

It follows that A is MDS. A similar construction works for $\mathbb{F}_p$.

Efficient Implementation. We refer to App. A for a complete description about possible strategies for efficient POSEIDON and STARKAD implementations.

3 Cryptanalysis Summary of STARKAD and POSEIDON Hashes

As for any new design, it is paramount to present a concrete security analysis. In the following, we provide an in-depth analysis of the security of our construction. Due to a lack of any method to ensure that an hash function based on a sponge construction is secure against all possible attacks, we base our argumentation on the following consideration. As we just recalled in the previous section, when the internal permutation $\mathcal{P}$ of an $N = c + r$ bit sponge function is modeled as a randomly chosen permutation, the sponge hash function is indifferentiable from a random oracle up to $2^{c/2}$ calls to $\mathcal{P}$. Thus, we choose *the numbers of rounds of the inner permutation case in order to guarantee security against any (secret-/known-/chosen-) distinguisher. Equivalently, this means that such number of rounds guarantee that $\mathcal{P}$ does not present any non-random/structural property (among the ones known in the literature*[11]*).*

Now we list the main points of our cryptanalysis results (which is given in details in the appendix). The number of rounds we can break depends on the security level M and the number of S-Boxes t, which we specify for each concrete hash function instance in the next section.

$\mathbb{F}_p^t$ *versus* $\mathbb{F}_{2^n}^t$. From the point of view of the designer, the prime field version $\mathbb{F}_p^t$ is always stronger than the binary field version $\mathbb{F}_{2^n}^t$, since fewer attacks apply. In particular, the designer must be taken into account the higher-order differential attack when he determines the number of rounds in order to guarantee security in $\mathbb{F}_{2^n}^t$. Vice-versa, this attack does not apply (or better, it is much less powerful) in $\mathbb{F}_p^t$ (due to the fact that the only subspaces of $\mathbb{F}_p$ are $\{0\}$ and the entire space).

Statistical Attacks. As we show in the following, the best statistical attacks (differential, linear, truncated/impossible differential attacks, rebound attack) cover at most 5 rounds with full S-Box layer for any S-box $(x^3, x^5, 1/x)$ we consider.

Algebraic Attacks. In order to estimate the security against algebraic attacks, we evaluate the degree of the reduced-round permutations and their inverses.

[11] We do not exclude that a non-random property can be discovered in the future.

Roughly speaking, our results can be summarized as following (where $n \simeq \log_2(p)$ for the prime field):

Interpolation Attack. The interpolation attack depends on the number of different monomials of the interpolation polynomial, where (an upper/lower bound of) the number of different monomials can be estimated given the degree of the function. The idea of such attack is to construct an interpolation polynomial that describes the function. If the number of monomials is too big, then such polynomial can not be constructed faster than via a brute force attack. We show that when the polynomial of degree D is dense, the attack complexity is $O(D^t)$, where a certain number of rounds must be added in order to guarantee that the polynomial becomes dense. For security level M bits $A_{Inter}(M,n,t)$ rounds can be attacked where

$$A_{Inter}(M,n,t) = \begin{cases} 0.63 \min\{n,M\} + \log_2 t, & S(x) = x^3 \\ 0.43 \min\{n,M\} + \log_2 t, & S(x) = x^5 \end{cases} \tag{1}$$

whereas for the S-box x^{-1} a more complicated equation must be satisfied for the attack to work:

$$R_P + \log_2 t R_F \leq 2 + \log_2(t) + \min\{M, \log_2(p)\} \tag{2}$$

Gröbner Basis. In a Gröbner basis attack, one tries to solve a system of non-linear equations that describe the function. The cost of such attack depend obviously on the degree of the equations, but also on the number of equations and on the number of variables. We show that the attack complexity is about $O(D^{2t})$, therefore for security level M bits the attack works at most on $\log_3 2^{\min\{n/2, M/2\}}$ rounds, which is smaller than for the interpolation attack. If a partial S-Box layer is used, it could become more efficient to consider degree-3 equations for single S-Boxes. In this case, more rounds can be necessary to guarantee security against this attack. With optimistic (for the adversary) complexity of the Gaussian elimination, we obtain for each S-box two attacks which are faster than 2^M if either condition is satisfied:

$$\underbrace{R_F + R_P}_{\text{number of rounds}} \leq 2 + \min(M,n) \cdot \begin{cases} 0.32, & S(x) = x^3 \\ 0.21, & S(x) = x^5 \\ 0.5 \log_2 t, & S(x) = x^{-1} \end{cases} \tag{3}$$

$$\underbrace{R_F + tR_P}_{\text{number of S-boxes}} \leq M \cdot \begin{cases} 0.18, & S(x) = x^3 \\ 0.14, & S(x) = x^5 \\ 0.25, & S(x) = x^{-1} \end{cases} \tag{4}$$

Higher-Order Differential. The higher-order differential attack depends on the *boolean degree*, where the boolean degree δ of a function $f(x) = x^d$ is given by $\delta = hw(d)$ where $hw(\cdot)$ is the hamming weight. The idea of such attack is based on the property that given a function $f(\cdot)$ of boolean degree δ, then

$\bigoplus_{x \in V \oplus \phi} f(x) = 0$ if the dimension of the subspace V satisfies $dim(V) \geq \delta + 1$. If the boolean degree is sufficiently high, then the attack does not work. The attack applies to the binary field case, where we use the x^3 S-box only. We obtained that the boolean degree grows accordingly to the algebraic degree, as the polynomial becomes dense and any monomial of degree d implies the existence of almost all monomials of smaller degree, which contain, among others, a monomial with degree of weight $\log_2 d$. Eventually we obtain the following condition for the attack to work in the binary field case:

$$R_F + R_P \leq 0.63 \min\{M, n\} + 2 + \log_2 t. \tag{5}$$

Zero-Sum Partition. The zero-sum partition distinguisher can be applied for $q = q_1 + q_2$ rounds as long as the boolean degree in the forward direction for q_1 and in the backward direction for q_2 does not exceed M. This allows attacking the same number of rounds as for the higher-order differential attack as the inverse function has high algebraic degree.

Security Margin. Given the *minimum* number of rounds necessary to guarantee security against all attacks known in the literature, we *arbitrary* decided by adding:

- two more rounds with full S-Box layer (+2 R_F);
- 7.5% more rounds with partial S-Box layer (+7.5% R_P).

4 Number of Rounds Needed for Security

The design goal is to offer a family of hash functions which minimize the R1CS costs (Starkad instances, Section 6) or AET (STARK) costs in (Poseidon instances, Section 7). It turns out that for the fixed S-box function the minimum costs are delivered by a primitive with the smallest number of S-boxes, though the field size also plays a role. For each combination (security level M, prime/binary field type, S-box size, S-box function) we minimize the number of S-boxes taking into account Equations (1),(2),(3),(4),(5).

4.1 Minimize "Number of S-Boxes"

In our design strategy, we always exploit the "Wide-Trail" strategy in order to guarantee security against statistical attacks. In other words, for this class of attacks, we limit to work with rounds with full S-Box layer in order to guarantee security. All our instances are secure against statistical attacks if

$$R_F^{stat} \geq 6.$$

In order to minimize the number of S-boxes for given n and t, the goal is to find the best ratio between R_P and R_F that minimizes

$$\textit{number of S-Boxes } = t \cdot R_F + R_P \tag{6}$$

where $t \geq 2$ and where the number of non-linear operations is proportional to the number of S-Boxes.

Overall the S-box type and the number of rounds should be chosen as follows:

– If you plan to use a binary field $\mathbb{F}_{2^n}$:
  - Use S-box x^3;
  - Select R_F to 6 or higher.
  - Select R_P that minimizes tR_F+R_P such that no inequation (1),(3),(4),(5) is satisfied.

– If you plan to use a prime field $\mathbb{F}_p$ and $GCD(q, p-1) = 1$ for $q = 3$ or $q = 5$:
  - Use S-box x^q;
  - Select R_F to 6 or higher.
  - Select R_P that minimizes $tR_F + R_P$ such that no inequation (1),(3),(4) is satisfied.

– If you plan to use a prime field $\mathbb{F}_p$ and $GCD(q, p-1) \neq 1$ for $q = 3$ or $q = 5$:
  - Use S-box x^{-1};
  - Select R_F to 6 or higher.
  - Select R_P that minimizes $tR_F + R_P$ such that no inequation (2),(3),(4) is satisfied.

We have set up a script that calculates the number of rounds accordingly. We added some security margin of at least 25% (sometimes more). Our resulting instances are given in Tables 2, 3, 4.

Results via Script. A complete analysis on how to set up the script – in order to guarantee security and to find the best ratio between R_P and R_F – for this case has been proposed in [32]. For this reason, we refer to [32], and we limit ourselves here to report the minimum number of rounds necessary to guarantee security.

For completeness, we mention that the simplest way to set up the script is to test (e.g. by brute force) all possible values R_P and R_F that guarantee security (equivalently, for which previous inequalities are satisfied), and finds the ones that minimize the (metric) cost.

4.2 Minimize "Number of S-Boxes × Field Size"

Secondly, we consider the metric given by "number of S-Boxes × field size", which well describes the cost of the Picnic PQ-Signature Scheme – where HadesCubic instantiated over $\mathbb{F}_{2^n}$ (low-data case). In this case, for each N and t, the goal is to find the best ratio of R_P and R'_F (where $R_F = R_F^{stat} + R'_F \geq R_F$) for which the following cost is minimized

$$n \times \big(t \cdot R_F + R_P\big) = N \cdot R_F + n \cdot R_P. \tag{7}$$

If both n and t are fixed, this metric is proportional to the one given before (that is, it is equal to the one given in (6) times a factor n). Thus, the results given in the previous section hold also for this metric. As before, for the case in which t is not fixed, we provide *a script that takes in input N and returns the best t and the best ratio between R_F and R_P that minimizes the metric given in (6).*

5 Concrete Instantiations – Poseidon$^\pi$ and Starkad$^\pi$

For our applications, we are interested in the cases:

- texts size: $N = 1\,536 = 3 \cdot 2^9$ (where $N = n \cdot t \simeq t \cdot \log_2 p$);
- security level: $M = 128$ and/or 256.

All our MDS matrices are Cauchy matrices, and the method to construct them is further described in Section 2.3. We use ascending sequences of integers (or elements in $\mathbb{F}_{2^n}$) for the construction.

The round constants are generated using the Grain LFSR [21] in a self-shrinking mode:

- Initialize the state with 80 bits $b_0, b_1, \ldots, b_{79}$ set to 1.
- Update the bits using $b_{i+80} = b_{i+62} \oplus b_{i+51} \oplus b_{i+38} \oplus b_{i+23} \oplus b_{i+13} \oplus b_i$.
- Discard the first 160 bits.
- Evaluate bits in pairs: If the first bit is a 1, output the second bit. If it is a 0, discard the second bit.

If a randomly sampled integer is not in $\mathbb{F}_p$, we discard this value and take the next one. Note that cryptographically strong randomness is not needed for the round constants, and other methods can also be used. We give both the matrices and the round constants in an auxiliary file for two example instantiations:

- x^3 – Poseidon-Permutation in $\mathbb{F}_p$ with $p = 2^{64} - 2^8 - 1$, $n = 64$, $t = 24$, $N = 1536$,
- x^{-1} – Poseidon-Permutation in $\mathbb{F}_p$ with $p = 27742317777372353535851937790883648493 + 2^{252}$, $n = 252$, $t = 6$, $N = 1512$,

Table 2: A range of different parameter sets for STARKAD$^{\pi}$ and POSEIDON$^{\pi}$ instantiated by S-Box$(x) = x^3$ (with security margin).

Security M	**Text Size** $N = n \times t$	**S-Box Size** (n or $\log_2 p$)	**# S-Boxes** (t)	$\mathbf{R_F}$	$\mathbf{R_P}$	**Field**	**Cost 1** Eq. (6)	**Cost 2** Eq. (7)
128	1536	768	2	10	81	$\mathbb{F}_p$	101	77568
128	1536	384	4	10	82	$\mathbb{F}_p$	122	46848
128	1536	256	6	8	84	$\mathbb{F}_p$	132	33792
128	1536	192	8	8	84	$\mathbb{F}_p$	148	28416
128	1536	96	16	8	64	$\mathbb{F}_p$	192	18432
128	1512	63	24	12	39	$\mathbb{F}_{2^n}$	327	20601
128	1551	33	47	12	20	$\mathbb{F}_{2^n}$	584	19272
128	1581	31	51	12	19	$\mathbb{F}_{2^n}$	631	19561
256	1536	768	2	12	166	$\mathbb{F}_p$	190	145920
256	1536	384	4	10	169	$\mathbb{F}_p$	209	80256
256	1536	256	6	10	169	$\mathbb{F}_p$	229	58624
256	1536	192	8	10	126	$\mathbb{F}_p$	206	39552
256	1536	96	16	8	64	$\mathbb{F}_p$	192	18432
256	1512	63	24	14	37	$\mathbb{F}_{2^n}$	373	23499
256	1551	33	47	14	18	$\mathbb{F}_{2^n}$	676	22308
256	1581	31	51	14	17	$\mathbb{F}_{2^n}$	731	22661

and

– x^3 – STARKAD-Permutation in $\mathbb{F}_{2^n}$ with $p(x) = x^{63} + x + 1$, $n = 63$, $t = 24$, $N = 1512$.

We also include reference implementations for various instantiations, and the code is available at https://extgit.iaik.tugraz.at/krypto/hadeshash.

6 SNARKs Application via POSEIDON$^{\pi}$

ZK-SNARKs [35,20] and Bulletproofs [11] are powerful proof systems to prove the computational integrity of very complex program executions. They are helpful in cryptographic protocols where Prover proves the knowledge of a hash function preimage or an opening in a Merkle tree. Such protocols are popular in crypto-currencies where they make possible to hide the transaction origin or amount by only proving it had been earlier included to a Merkle tree. Both SNARKs and Bulletproofs work with programs represented as arithmetic circuits over some prime field $GF(p)$.

In SNARKs, the prime field is typically the scalar field of some point on a pairing-friendly elliptic curve, whereas in Bulletproofs the curve does not have to be pairing-friendly (thus fast curves such as Curve25519 are a popular choice). The primitive POSEIDON$^{\pi}$ can be represented as such circuit with reasonably few

Table 3: A range of different parameter sets for POSEIDON$^\pi$ instantiated by S-Box$(x) = x^5$ (with security margin).

Security M	**Text Size** $N = n \times t$	**S-Box Size** (n or $\log_2 p$)	**# S-Boxes** (t)	$\mathbf{R_F}$	$\mathbf{R_P}$	**Field**	**Cost 1** Eq. (6)	**Cost 2** Eq. (7)
128	1536	768	2	8	56	$\mathbb{F}_p$	72	55296
128	1536	384	4	8	56	$\mathbb{F}_p$	88	33792
128	1536	256	6	8	57	$\mathbb{F}_p$	105	26880
128	1536	192	8	8	57	$\mathbb{F}_p$	121	23232
128	1536	96	16	8	42	$\mathbb{F}_p$	170	16320
256	1536	768	2	10	113	$\mathbb{F}_p$	133	102144
256	1536	384	4	8	116	$\mathbb{F}_p$	148	56832
256	1536	256	6	8	117	$\mathbb{F}_p$	165	42240
256	1536	192	8	8	86	$\mathbb{F}_p$	150	28800
256	1536	96	16	8	42	$\mathbb{F}_p$	170	16320

Table 4: A range of different parameter sets for POSEIDON$^\pi$ instantiated by S-Box$(x) = x^{-1}$ (with security margin).

Security M	**Text Size** $N = n \times t$	**S-Box Size** (n or $\log_2 p$)	**# S-Boxes** (t)	$\mathbf{R_F}$	$\mathbf{R_P}$	**Field**	**Cost 1** Eq. (6)	**Cost 2** Eq. (7)
128	1536	768	2	8	135	$\mathbb{F}_p$	151	115968
128	1536	384	4	8	129	$\mathbb{F}_p$	161	61824
128	1536	256	6	8	127	$\mathbb{F}_p$	175	44800
128	1536	192	8	8	124	$\mathbb{F}_p$	188	36096
128	1536	96	16	8	84	$\mathbb{F}_p$	212	20352
256	1536	768	2	50	100	$\mathbb{F}_p$	200	153600
256	1536	384	4	8	267	$\mathbb{F}_p$	299	114816
256	1536	256	6	8	265	$\mathbb{F}_p$	313	80128
256	1536	192	8	8	193	$\mathbb{F}_p$	257	49344
256	1536	96	16	8	84	$\mathbb{F}_p$	212	20352

gates, but the parameters of POSEIDON$^\pi$ must have been determined first by p. Concretely, after p is fixed, we first check if x^3 or x^5 are bijections in $GF(p)$, which is true if $p \bmod 3 \neq 1$ (resp., $p \bmod 5 \neq 1$). If both inequalities are not satisfied, we have to use the inverse S-Box.

Complexity in SNARKs Applications. The SNARK prover complexity is $O(s)$ where s is the number of rank-1 constraints – quadratic equations of form $(\sum_i u_i X_i)(\sum_i v_i X_i) = \sum_i w_i X_i$ where u_i, v_i, w_i are field elements and X_i are program variables. It is easy to see that the S-Box x^3 is represented by 2 constraints, the S-Box x^5 by 3 constraints, and the S-Box $1/x$ by 3 constraints

(1 for non-zero case, and two more for the zero case). Thus in total we have

$$2tR_F + 2R_P \text{ constraints for } x^3\text{-based } \textsc{Poseidon}^\pi; \tag{8}$$

$$3tR_F + 3R_P \text{ constraints for } x^5\text{-based } \textsc{Poseidon}^\pi; \tag{9}$$

$$3tR_F + 3R_P \text{ constraints for } x^{-1}\text{-based } \textsc{Poseidon}^\pi. \tag{10}$$

It requires a bit more effort to see that we do not need more constraints as the linear layers and round constants can be incorporated into these ones. However, it is necessary to do some preprocessing. For example, in the $\textsc{Poseidon}^\pi$ setting the full S-Box layers are followed by linear transformation $M = (M_{i,j})$. Each round with full S-Box can be represented by the following constraints in the SNARK setting.

$$\left(\sum_j M_{i,j} x_{i,j}\right) \cdot \left(\sum_j M_{i,j} x_{i,j}\right) = y_i \;\; 1 \leq i \leq t \tag{11}$$

$$y_i \cdot \left(\sum_j M_{i,j} z_{i,j}\right) = z_i \tag{12}$$

where $M = I_{t\times t}$ for the first round. However, in a round with partial S-Box layer we will have only one such constraint for $j = 1$. For the rest of the $t - 1$ variables we will have linear constraints of the form

$$\sum_j M_{i,j} x_{i,j} = u_i \;\; \text{where} 2 \leq i \leq t.$$

Since the linear constrains have no role in the SNARK, in the following partial S-Box rounds the linear constraints can be composed with (from the previous round(s)) using following equation

$$\sum_k M_{i,k} \left(\sum_j M_{i,j} x_{i,j}\right) = v_k \;\; 2 \leq k \leq t$$

We can now calculate the number of constraints for the sponge-based hash functions and Merkle trees. In sponges, the $2M$ bits are reserved for the capacity, so $N - 2M$ bits are fed with message. Therefore, we get

- $\frac{2tR_F+2R_P}{N-2M}$ constraints per bit for x^3-based $\textsc{Poseidon}^\pi$;
- $\frac{3tR_F+3R_P}{N-2M}$ constraints per bit for x^5-based $\textsc{Poseidon}^\pi$;
- $\frac{3tR_F+3R_P}{N-2M}$ constraints per bit for x^{-1}-based $\textsc{Poseidon}^\pi$.

Similarly we obtain that the Merkle tree based on such a sponge function has branching $\frac{N}{2M} - 1$. Based on that we can calculate how many constraints we need to prove the opening in a Merkle tree of, for example, 2^{32} elements (the recent ZCash setting). The tree will have $32 \log_{-1+N/2M} 2$ levels with the number of constraints in each according to the above.

7 STARKs Application via $\textsc{Starkad}^\pi$

ZK-STARKs [6] is another generic proof system for the computational integrity. STARKs operate with programs, whose internal state can be represented as a set of w registers, each belonging to a binary field $GF(2^n)$ (or extension of it). Here $n = 32$ and higher are preferable. The program execution is then represented as a set of T internal states. The computational integrity is defined as the set of all wT registers satisfying certain s polynomial equations (constraints) of degree d.

STARKs Costs. According to [28], the number of constraints does not play a major role in the prover, verifier, or communication complexity, which are estimated as follows:

$$\text{Prover Operations in } GF(2^n) = 8w \cdot T \cdot d \cdot \log(wT) \tag{13}$$

$$\text{Prover Memory} = \Omega(w \cdot T \cdot n) \tag{14}$$

$$\text{Communication} = \text{Verifier Time} = n \cdot (m + \log^2(8Td)) \tag{15}$$

where m is the maximum number of variables in a constraint polynomial.

The primitive $\textsc{Starkad}^\pi$ can be represented as such program with few registers and number of steps and low degree. For x^3 to be invertible n has to be odd, so we select $n = 63$ for our primary instance of $\textsc{Starkad}^\pi$ to be close to 64 bits to be able to efficiently utilize the carry-less multiplication (CLMUL) instruction-set available in recent CPUs to speed up finite field operations. Following the same approach as for SNARKs in Section 6, we keep in registers only S-Box inputs and the permutation outputs. Setting $w = t$, we get $T = R_F + \lceil R_P/t \rceil$. Thus the complexity is calculated as follows:

$$\text{Prover Operations in } GF(2^n) = 24(tR_F + R_P) \cdot \log(tR_F + R_P) \tag{16}$$

$$\text{Prover Memory} = \Omega(63 \cdot (tR_F + R_P)) \tag{17}$$

$$\text{Communication} = \text{Verifier Time} = 63 \cdot (t + \log^2(24(tR_F + R_P))) \tag{18}$$

We are flexible in choosing the number of S-Boxes t. For the uniformity, the permutation width of about 1550 bits yields $t = 24$, which requires to set $R_F = 12, R_P = 34$ to protect from attacks and have reasonable security margin. In the sponge setting, we reserve 2 S-Boxes for the capacity in the 128-bit security level and 4 S-Boxes for the capacity in the 256-bit security level. Thus for our primary instance $\textsc{Starkad} - 128$ we get 67 675 operations in $GF(2^{63})$ for each permutation call. As we process 1260 bits per call, we obtain the prover complexity of 54 operations per bit.

In the Merkle tree setting, the rate/capacity ratio in $\textsc{Starkad} - 128$ is 5, which means that the Merkle tree has branching 5. The number of prover operations to prove the opening in a tree with Q elements is then $67\,675 \log_5 N$.

AET complexity of Friday. Friday [3] is a recent STARK-friendly symmetric hash function introduced by Ashur and Dhooghe. It is presented in several instances with different security levels. Friday-128 offers 64 bits of collision resistance, and Friday-256 offers 128 bits of collision resistance. Friday is a blockcipher in the MP mode of operation where the chaining value serves as a key.

Friday-128 utilizes 10 (14 for Friday-256) rounds of the following structure. An inverse S-Box (in $GF(2^{128}), GF(2^{256})$) is followed by two transformations of degree 4 in the field and then a constant addition. We have to use 5 registers per round: S-Box input, S-Box output, a temporary register to store the information if the input is zero, the output of the first degree-4 transformation, and the round constant. All these variables are linked by constraints of degree not more than 4. One can optimize it by adding two more intermediate variables for the linear transformations and reducing the degree to 2. In total, we get $T = 10(14), w = 7, d = 2$:

$$\text{Prover Operations in Friday-128} = 8 \cdot 7 \cdot 2 \cdot 10 \log 70 = 6865 \tag{19}$$

$$\text{Prover Operations in Friday-256} = 8 \cdot 7 \cdot 2 \cdot 14 \log 98 = 10372 \tag{20}$$

or 54 (respectively, 41) operation per bit. Note that these operations are done in bigger fields than our $GF(2^{63})$ so the actual time difference is much bigger.

Acknowledgements

This work is partially supported by the Ethereum foundation, Starkware Ltd, and IOV42 Ltd.

References

1. Abdelraheem, M.A., Ågren, M., Beelen, P., Leander, G.: On the Distribution of Linear Biases: Three Instructive Examples. In: CRYPTO 2012. LNCS, vol. 7417, pp. 50–67 (2012)
2. Albrecht, M.R., Grassi, L., Rechberger, C., Roy, A., Tiessen, T.: MiMC: Efficient Encryption and Cryptographic Hashing with Minimal Multiplicative Complexity. In: ASIACRYPT 2016. LNCS, vol. 10031, pp. 191–219 (2016)
3. Ashur, T., Dhooghe, S.: Marvellous: a stark-friendly family of cryptographic primitives. Cryptology ePrint Archive, Report 2018/1098 (2018), https://eprint.iacr.org/2018/1098
4. Bardet, M., Faugere, J., Salvy, B., Yang, B.: Asymptotic behaviour of the index of regularity of quadratic semi-regular polynomial systems. In: The Effective Methods in Algebraic Geometry Conference (MEGA). pp. 1–14 (2005)
5. Beierle, C., Canteaut, A., Leander, G., Rotella, Y.: Proving Resistance Against Invariant Attacks: How to Choose the Round Constants. In: CRYPTO 2017. LNCS, vol. 10402, pp. 647–678 (2017)

6. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Scalable, transparent, and post-quantum secure computational integrity. IACR Cryptology ePrint Archive **2018**, 46 (2018)
7. Bertoni, G., Daemen, J., Peeters, M., Assche, G.V.: On the Indifferentiability of the Sponge Construction. In: EUROCRYPT 2008. LNCS, vol. 4965, pp. 181–197 (2008)
8. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. Journal of Cryptology **4**(1), 3–72 (1991)
9. Biham, E., Shamir, A.: Differential Cryptanalysis of the Data Encryption Standard. Springer (1993)
10. Boura, C., Canteaut, A., De Cannière, C.: Higher-Order Differential Properties of Keccak and *Luffa*. In: FSE 2011. LNCS, vol. 6733, pp. 252–269 (2011)
11. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: IEEE Symposium on Security and Privacy. pp. 315–334. IEEE Computer Society (2018)
12. Camenisch, J., Kohlweiss, M., Soriente, C.: An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In: Public Key Cryptography. Lecture Notes in Computer Science, vol. 5443, pp. 481–500. Springer (2009)
13. Camenisch, J., Lysyanskaya, A.: Dynamic accumulators and application to efficient revocation of anonymous credentials. In: Yung, M. (ed.) Advances in Cryptology — CRYPTO 2002. pp. 61–76. Springer Berlin Heidelberg, Berlin, Heidelberg (2002)
14. Carlet, C., Charpin, P., Zinoviev, V.A.: Codes, Bent Functions and Permutations Suitable For DES-like Cryptosystems. Designs, Codes Cryptography **15**(2), 125–156 (1998)
15. Cox, D.A., Little, J., O'Shea, D.: Ideals, varieties, and algorithms - an introduction to computational algebraic geometry and commutative algebra (2. ed.). Undergraduate texts in mathematics, Springer (1997)
16. Daemen, J., Knudsen, L.R., Rijmen, V.: The Block Cipher Square. In: FSE 1997. LNCS, vol. 1267, pp. 149–165 (1997)
17. Dinur, I., Kales, D., Promitzer, A., Ramacher, S., Rechberger, C.: Linear Equivalence of Block Ciphers with Partial Non-Linear Layers: Application to LowMC. In: EUROCRYPT 2019. p. to appear. LNCS, Springer (2019)
18. Grassi, L.: Mixture Differential Cryptanalysis: a New Approach to Distinguishers and Attacks on round-reduced AES. IACR Trans. Symmetric Cryptol. **2018**(2), 133–160 (2018)
19. Grassi, L., Rechberger, C., Rønjom, S.: A New Structural-Differential Property of 5-Round AES. In: EUROCRYPT 2017. LNCS, vol. 10211, pp. 289–317 (2017)
20. Groth, J.: On the size of pairing-based non-interactive arguments. In: EUROCRYPT (2). Lecture Notes in Computer Science, vol. 9666, pp. 305–326. Springer (2016)
21. Hell, M., Johansson, T., Maximov, A., Meier, W.: The grain family of stream ciphers. In: The eSTREAM Finalists, Lecture Notes in Computer Science, vol. 4986, pp. 179–190. Springer (2008)
22. Hopwood, D., Bowe, S., Hornby, T., Wilcox, N.: Zcash protocol specification: Version 2019.0-beta-37 [overwinter+sapling]. Tech. rep., Zerocoin Electric Coin Company (2019), available at https://github.com/zcash/zips/blob/master/protocol/protocol.pdf
23. Jakobsen, T., Knudsen, L.R.: The Interpolation Attack on Block Ciphers. In: FSE 1997. LNCS, vol. 1267, pp. 28–40 (1997)
24. Jean, J., Naya-Plasencia, M., Peyrin, T.: Multiple Limited-Birthday Distinguishers and Applications. In: SAC 2013. LNCS, vol. 8282, pp. 533–550 (2013)

25. Knudsen, L.R.: Truncated and Higher Order Differentials. In: FSE 1994. LNCS, vol. 1008, pp. 196–211 (1994)
26. Lamberger, M., Mendel, F., Rechberger, C., Rijmen, V., Schläffer, M.: Rebound Distinguishers: Results on the Full Whirlpool Compression Function. In: ASIACRYPT 2009. LNCS, vol. 5912, pp. 126–143 (2009)
27. Leander, G., Abdelraheem, M.A., AlKhzaimi, H., Zenner, E.: A Cryptanalysis of PRINTcipher: The Invariant Subspace Attack. In: CRYPTO 2011. LNCS, vol. 6841, pp. 206–221 (2011)
28. Ltd, S.I.: The complexity of stark-friendly cryptographic primitives. Private communication (2018)
29. MacWilliams, F.J., Sloane, N.J.A.: The Theory of Error-Correcting Codes. North-holland Publishing Company (1978)
30. Matsui, M.: Linear Cryptanalysis Method for DES Cipher. In: EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397 (1993)
31. Mendel, F., Rechberger, C., Schläffer, M., Thomsen, S.S.: The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl. In: FSE 2009. LNCS, vol. 5665, pp. 260–276 (2009)
32. *No Authors Given*: On a Generalization of Substitution-Permutation Networks: The HADES Design Strategy, in Submission
33. Nyberg, K.: Differentially uniform mappings for cryptography. In: EUROCRYPT 1993. LNCS, vol. 765, pp. 55–64 (1994)
34. Nyberg, K., Knudsen, L.R.: Provable Security Against Differential Cryptanalysis. In: CRYPTO 1992. LNCS, vol. 740, pp. 566–574 (1992)
35. Parno, B., Howell, J., Gentry, C., Raykova, M.: Pinocchio: Nearly practical verifiable computation. In: IEEE Symposium on Security and Privacy. pp. 238–252. IEEE Computer Society (2013)
36. Rijmen, V., Daemen, J., Preneel, B., Bosselaers, A., Win, E.D.: The cipher SHARK. In: Fast Software Encryption – FSE 1996. LNCS, vol. 1039, pp. 99–111. Springer (1996)
37. Youssef, A.M., Mister, S., Tavares, S.E.: On the Design of Linear Transformations for Substitution Permutation Encryption Networks. In: School of Computer Science, Carleton University. pp. 40–48 (1997)

SUPPLEMENTARY MATERIAL

A Efficient Implementation

Like for LowMC, the fact that the non-linear layer is partial in R_P rounds can be used to reduce the size of the round constants required in each round R_P. Referring to [17], we recall here an equivalent representation of an SPN with partial non-linear layer for an efficient implementation.

Round Constants. In the description of an SPN, it is possible to swap the order of the linear layer and the round constant addition as both operations are linear. The round constant then needs to be exchanged with an equivalent one. For round constant $c^{(i)}$, the equivalent one can be written as $\hat{c}^{(i)} = MC^{-1}(c^{(i)})$, where MC is the linear layer in the i-th round. If one works with partial non-linear layers, it is possible to use this property to move parts of the original round constants from the last round all the way through the permutation to the beginning. To arrive at such a reduced variant, we work as following:

- First, we find an equivalent round constant that is applied before the affine layer.
- Then we split the round constants in two parts, one that applies to the S-Box part of the non-linear layer and one that applies to the identity part of the non-linear layer. The constant part that only applies to the non-linear layer part can now move further up.
- Working in this way for all round constants, we finally end up with an equivalent representation in which round constants are only added to the output of the S-Boxes apart from one constant which is applied to the entire state after the first R_f rounds.

This simplified representation can in certain cases also reduce the implementation cost of an SPN permutation with a partial non-linear layer. For instance, the standard representation of HadesCubic requires constants matrices of total size $t \cdot n \cdot (R+1)$, where $R = R_P + R_F$ is the number of rounds. The optimized representation only requires $t \cdot n \cdot (R_F + 1) + n \cdot R_P$, thus potentially greatly reducing the amount of needed memory and calculation to produce the round constants.

Linear Layer. For our design the situation is simpler than for LowMC, since we can guarantee the existence of invertible sub matrices. Hence, a similar trick can be used also for the matrix multiplication.

Focusing on the rounds with a single S-Box, let M be the $t \times t$ MDS matrix of the linear layer:

$$M = \left[\begin{array}{c|ccccc} M_{0,0} & M_{0,1} & M_{0,2} & \cdots & M_{0,t-1} & M_{0,t} \\ \hline M_{1,0} & & & & & \\ M_{2,0} & & & & & \\ \vdots & & & \hat{M} & & \\ M_{t-1,0} & & & & & \\ M_{t,0} & & & & & \end{array}\right] \equiv \left[\begin{array}{c|c} M_{0,0} & v \\ \hline w & \hat{M} \end{array}\right]$$

where $\hat{M}$ is a $(t-1) \times (t-1)$ MDS matrix (note that since M is MDS, every submatrix of M is also MDS), v is a $1 \times (t-1)$ matrix and w is a $(t-1) \times 1$ vector. By simple computation, the following equivalence holds:

$$M = \underbrace{\left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & \hat{M} \end{array}\right]}_{M'} \times \underbrace{\left[\begin{array}{c|c} M_{0,0} & v \\ \hline \hat{w} & I \end{array}\right]}_{M''}, \tag{21}$$

where

$$\hat{w} = \hat{M}^{-1} \times w$$

and I is the $(t-1) \times (t-1)$ identity matrix. Note that both M' and $M^{''}$ are two invertible matrices[12].

As for the round constants discussed previously, it is possible to use the equivalence (21) in order *to swap the S-Box layer (formed by a single S-Box and $t-1$ identity functions) and the matrix multiplication with the matrix M'*. As a result, each linear part in the R_P rounds is defined only by a multiplication with a matrix of the form $M^{''}$, which is a *sparse matrix*, since $(t-1)^2 - (t-1) = t^2 - 3t + 2$ coefficients of $M^{''}$ are equal to zero (moreover, $t-1$ coefficients of $M^{''}$ are equal to one). It follows that this optimized representation – potentially – greatly reduces the amount of needed memory and calculation to compute the linear layer multiplication.

B Security Analysis – Starkad and Poseidon with S-Box$(x) = x^3$

B.1 Security Analysis - Statistical Attacks

Differential Cryptanalysis. Differential cryptanalysis [8,9] and its variations are the most widely used techniques to analyze symmetric-key primitives. The differential probability of any function over the finite field $\mathbb{F}_{2^n}$ is defined as

$$Prob[\alpha \to \beta] := |\{x : f(x) \oplus f(x \oplus \alpha) = \beta\}|/(2^n).$$

[12] First of all, $\det(M') = \det(\hat{M}) \neq 0$ since $\hat{M}$ is an MDS matrix, and so it is invertible. Secondly, $\det(M) = \det(M') \cdot \det(M^{''})$. Since $\det(M) \neq 0$ and $\det(M') \neq 0$, it follows that $\det(M^{''}) \neq 0$.

Since the cubic function $f(x) = x^3$ is an almost perfect non-linear permutation (APN) [34,33], it has an optimal differential probability over a prime field or $\mathbb{F}_{2^n}$ (where n is odd). In other words, for this function the probability is bounded above by $2/2^n$ or $2/|\mathbb{F}_p|$.

As largely done in the literature, we claim that Starkad and Poseidon are secure against differential cryptanalysis if each characteristic has probability at most 2^{-N}.

In order to compute the minimum number of rounds to guarantee this, we work only with the rounds with full S-Box layers. In other words, we limit ourselves to work with a "weaker" version of the permutation defined as

$$R^{R_f} \circ L \circ R^{R_f}(\cdot), \tag{22}$$

where

- L is an *invertible linear layer* (which is the "*weakest*" possible assumption),
- $R(\cdot) = M \circ \text{S-Box} \circ ARK(\cdot)$ where S-Box$(\cdot)$ is a full S-Box layer (remember that M is an MDS matrix).

We are going to show that this "weaker" permutation is secure against differential cryptanalysis for $R_F = 2R_f = 6$ if $t + 2 < 2n$, and $R_F = 8$ otherwise. As a result, it follows that also Starkad/Poseidon (instantiated with R_F rounds with full S-Box layers) is secure against such an attack. Indeed, if the linear layer L (which we only assume to be invertible) is replaced by R_P rounds of Starkad/Poseidon, its security cannot decrease. *The same strategy is exploited in the following in order to prove security against all attacks in this subsection.*

In order to prove the result just given, we need a lower bound on the number of minimum number of active S-Boxes. Observe that the minimum number of "active" S-Boxes in the permutation

$$R^s \circ L \circ R^r(\cdot) \equiv SB \circ \underbrace{M \circ SB}_{s-1 \text{ times}} \circ \underbrace{L'}_{\equiv L \circ M(\cdot)} \circ SB \circ \underbrace{M \circ SB}_{r-1 \text{ times}}(\cdot)$$

(where $s, r \geq 1$, $R(\cdot)$ is a round with full S-Box layer and where L' is an invertible linear layer) are at least[13]

$$\text{number } \textit{active} \text{ S-Boxes} \geq \underbrace{\big(\lfloor s/2 \rfloor + \lfloor r/2 \rfloor\big) \cdot (t+1)}_{\text{due to final/initial rounds}} + \big(s \bmod 2\big) + \big(r \bmod 2\big).$$

We emphasize that the (middle) linear $L'(\cdot) \equiv L \circ M(\cdot)$ plays *no* role in the computation of the previous number. Since at least $2 \cdot (t+1) + 1$ S-Boxes are active in the 5 middle rounds of $R^r \circ L \circ R^{5-r}(\cdot)$ for $1 \leq r \leq 4$, and since the

[13] If $s = 2 \cdot s'$ is even, then the minimum number of active S-Boxes over $R^s(\cdot)$ rounds with full S-Box layer is $\lfloor s/2 \rfloor \cdot (t+1)$. Instead, if $s = 2 \cdot s' + 1$ is odd, then the minimum number of active S-Boxes over $R^s(\cdot)$ rounds with full S-Box layer is $\lfloor s/2 \rfloor \cdot (t+1) + 1$.

maximum differential probability of the cubic S-Box is $DP_{max} = 2^{-n+1}$, each characteristic has probability at most

$$(2^{-n+1})^{2\cdot(t+1)+1} = 2^{-N} \cdot 2^{-N-3n+2t+3} < 2^{-N},$$

since $[N + 3n = n \cdot (t + 3)] > [2t + 3 = 2 \cdot (t + 3/2)]$, where $t + 3 > t + 3/2$ and $n \geq 3$. Finally, 1 more round guarantees that no differential attack can be set up.

Similarly, in the case in which $t + 2 < 2n$, it is sufficient to consider the 3 middle rounds to guarantee security against differential cryptanalysis. Indeed, each characteristic has probability $(2^{-n+1})^{t+2} = 2^{-N} \cdot 2^{-2n+t+2} < 2^{-N}$, since at least $t+2$ S-Boxes are active. Again, 1 more round guarantees that no differential attack can be set up.

Security up to $2^M \leq 2^N$*.* For completeness, we present the number of rounds necessary to provide security up to 2^M (that is, data and computational cost of the attacker upper bounded by 2^M). Using the same analysis as before, it turns out that

$$R_F = \begin{cases} 4 & \text{if } t + 2 < N + 2n - M \\ 6 & \text{if } t + 2 \geq N + 2n - M \end{cases}$$

guarantees that no differential attack can be set up.

Linear Cryptanalysis. Similar to differential attacks, linear attacks [30] pose no threat to the Starkad/Poseidon families of permutations instantiated with the same number of rounds previously defined for classical differential cryptanalysis. This follows from the fact that the cubic function is almost bent (AB), which means that its maximum square correlation is limited to 2^{-n+1} (see [1] for details). As a result, it offers the best possible resistance against linear cryptanalysis much like an APN function provides optimal resistance against differential cryptanalysis.

For completeness, we remember a function $f(\cdot)$ is AB and/or APN if and only if its inverse $f^{-1}(\cdot)$ is AB and/or APN [14]. As a result, both the forward and the inverse permutation are secure against linear and differential cryptanalysis[14].

Truncated Differential. A variant of classical differential cryptanalysis is the truncated differential one [25], in which the attacker can specify only part of the difference between pairs of texts.

We consider the "weaker" permutation described in (22) again. Focusing only on active/passive bytes (and not on the actual differences), there exist several differentials with probability 1 for a maximum of 1 round of Starkad/Poseidon,

[14] Remember that if a matrix M is MDS, then also M^{-1} is MDS.

e.g.

$$[\alpha, 0, ..., 0]^T \xrightarrow{R(\cdot)} M \times [\beta, 0, ..., 0]^T$$

where α, β denote non-zero differences. Due to the next S-Box layer, the linear relations given by $M \times (\beta, 0, ..., 0)^T$ are destroyed in the next round. As a result, no probability-one truncated differential covers more than a single round.

Since no linear relation survives the S-Box layer, it seems hard to set up a truncated differential for more than 2 rounds. As a result, it turns out that 4 rounds with full S-Box layer makes HadesCubic$^\pi$ secure against this attack.

Rebound Attacks. The rebound attacks [26,31] have much improved the best known attacks on many hash functions, especially for AES-based schemes. The goal of this attack is to find two (input, output) pairs (p^1, c^1) and (p^2, c^2) such that the two inputs satisfy a certain (truncated) input difference and the corresponding outputs satisfy a certain (truncated) output difference.

The rebound attack consists of two phases, called *inbound* and *outbound* phase. According to these phases, the internal permutation of the hash function is split into three sub-parts. Let f be the permutation, then we get $f = f_{fw} \circ f_{in} \circ f_{bw}$. The part of the inbound phase is placed in the middle of the permutation and the two parts of the outbound phase are placed next to the inbound part. In the outbound phase, two high-probability (truncated) differential trails are constructed, which are then connected in the inbound phase. Since the rebound attack is a differential attack, as first thing an attacker needs to construct a *"good" (truncated) differential trail*. A good trail used for a rebound attack should have a high probability in the outbound phases and can have a rather low probability in the inbound phase. In the first phase, the attacker uses the knowledge of the key to find pairs of texts that satisfy the middle rounds of the truncated differential trail. In the second one, they propagate the solutions found in the first phase in the forward and in the backward directions, and check if at least one of them satisfies the entire differential trail.

The best rebound attack on AES proposed in [24] covers 8 rounds. Here we claim that 6 rounds with full S-Box layers are sufficient to protect Starkad/Poseidon from this attack. To support it, note that *(1st)* 1 round of Starkad/Poseidon provides full diffusion while 2 rounds of AES are necessary to provide it and *(2nd)* the best truncated differential covers 1 round of Starkad/Poseidon *vs* 3 rounds of AES[15]. Since the best results on AES in the literature cover at most 8 rounds, due to the similarity between AES and Starkad/Poseidon and due to the previous observations, we argue that it is not possible to mount a rebound attack on more than 5 rounds with full S-Box layers of Starkad/Poseidon. Hence, 6 rounds of Starkad/Poseidon with full S-Box layers are sufficient to guarantee security against this attack.

[15] The best truncated differential distinguisher with prob. 1 covers 2 rounds of AES.

Multiple-of-n and Mixed Differential Cryptanalysis. The "Multiple-of-8" distinguisher [19] was proposed at Eurocrypt 2017 by Grassi *et al.* as the first 5-round secret-key distinguisher for AES that exploits a property which is independent of the secret key and of the details of the S-Box. It is based on a new structural property for up to 5 rounds of AES: by appropriate choices of a number of input pairs it is possible to make sure that the number of times that the difference of the resulting output pairs lie in a particular subspace is always a multiple of 8. The input pairs of texts that satisfy a certain output difference are related by linear/differential relations. Such relations are exploited by a variant of such a distinguisher, called the "mixture differential" distinguisher [18] proposed at FSE/ToSC 2019.

Regarding Starkad/Poseidon, it is possible to set up such distinguishers on 2 rounds only. In particular, consider a set of texts with $2 \leq s \leq t$ active words (and $t - s$ constants words). The number of pairs of texts that satisfy an (arbitrary) output truncated differential is always a multiple of 2^{s-1}. Moreover, the relations of the input pairs of texts exploited by mixture differential cryptanalysis are known.

The proofs of these two properties are analogous to the ones proposed in [19] and in [18]. E.g., consider two texts T^1 and T^2 of the form

$$T^1 = C \oplus \left[x_0 \; x_1 \; 0 \; ... \; 0\right]^T, \qquad T^2 = C \oplus \left[y_0 \; y_1 \; 0 \; ... \; 0\right]^T$$

for some constant C and where $x_i \neq y_i$ for $i = 0, 1$. After one round, the difference in each word is of the form

$$M_0 \cdot [\text{S-Box}(x_0 \oplus c_0) \oplus \text{S-Box}(x_1 \oplus c_1)] \oplus M_1 \cdot [\text{S-Box}(y_0 \oplus c_0) \oplus \text{S-Box}(y_1 \oplus c_1)],$$

where M_0, M_1 depend on the MixLayer and c_0, c_1 depend on the secret key. By simple observation, the same output difference is given by the pair of texts

$$\hat{T}^1 = C \oplus \left[y_0 \; x_1 \; 0 \; ... \; 0\right]^T, \qquad \hat{T}^2 = C \oplus \left[x_0 \; y_1 \; 0 \; ... \; 0\right]^T.$$

Combining this result with a 1-round truncated differential with prob. 1, it is possible to set up a multiple-of-n distinguisher (where $n = 2^{s-1}$) and a mixture differential one on 2 rounds of Starkad/Poseidon. Using the inside-out approach, it is possible to set up such attack on 4-round of Starkad/Poseidon. As a result, it turns out that 6 rounds with full S-Box layers make it secure against these attacks.

Invariant Subspace Attack. The invariant subspace attack [27] makes use of affine subspaces that are invariant under the round function. As the round constant addition translates this invariant subspace [5], random round constants provides a good protection against such attacks.

Integral/Square Attack. Integral cryptanalysis is a technique first applied on SQUARE [16] and is particularly efficient against designs based on substitution-permutation networks, like AES or Starkad/Poseidon.

The idea is to study the propagation of sums of values. For the case of Starkad/Poseidon, it is possible to set up an integral distinguisher over two rounds, e.g.

$$\begin{bmatrix} A \\ C \\ \dots \\ C \end{bmatrix} \xrightarrow{\text{S-Box}(\cdot)} \begin{bmatrix} A \\ C \\ \dots \\ C \end{bmatrix} \xrightarrow{M(\cdot)} \begin{bmatrix} A \\ A \\ \dots \\ A \end{bmatrix} \xrightarrow{\text{S-Box}(\cdot)} \begin{bmatrix} A \\ A \\ \dots \\ A \end{bmatrix} \xrightarrow{M(\cdot)} \begin{bmatrix} B \\ B \\ \dots \\ B \end{bmatrix}$$

where A denotes an active word, C a constant one and B a balanced one[16]. Using the inside-out approach, it is possible to set up such attack on 4-round of HadesCubic$^\pi$. As a result, it turns out that 6 rounds with full S-Box layers make HadesCubic$^\pi$ secure against this attack.

B.2 Security Analysis - Algebraic Attacks

First we introduce a simple lemma, which follows from the iterative structure of the HadesCubic permutation.

Lemma 1. *The algebraic degree $D_3(r)$ of r-round* Starkad/Poseidon *with S-box x^3 as a function of input and, optionally, key variables is at most 3^r, no matter if partial or full rounds are used.*

Interpolation Attack. One of the most powerful attacks is the interpolation attack, introduced by Jakobsen and Knudsen [23] in 1997. In the case of a keyed function, the strategy of the attack is to construct a polynomial representation of the function without knowledge of the secret key. If an adversary can construct such a polynomial then it can compute any output without knowing the key, thus enabling forgeries (for MAC settings) and other attacks.

Let $E_k : \mathbb{F}_{2^N} \to \mathbb{F}_{2^N}$ be a keyed function. The interpolation polynomial $P(x)$ representing $E_k(x)$ can be constructed using e.g. the Vandermonde matrix - *cost* approximately of $\mathcal{O}(t^2)$ - or the Lagrange's theorem - *cost* approximately of $\mathcal{O}(t \cdot \log t)$, where x is the indeterminate corresponding to the input.

In more details, each output word of an SPN permutation can be represented as a multivariate polynomial where the variables are the inputs to each S-Box. *Consider a keyed permutation input where χ input words are unknown/variables to us, and the other $t - \chi$ words are known/fixed*:

$$\chi \text{ variables input words} \qquad \text{and} \qquad t - \chi \text{ fixed input words.}$$

[16] For completeness, we recall that given a set of texts $\{x_i\}_{i\in I}$, the word x^j is *active* if $x^j_i \neq x^j_l$ for each $i \neq l$, constant if $x^j_i = x^j_l$ for each i, l, and balanced if $\bigoplus_i x^j_i = 0$.

A (rough) estimation of the number of monomials of the interpolation polynomial (and so of the complexity of the attack) is given by

$$(D_3(r) + 1)^{\chi},$$

As a result, by requiring that the number of monomials be close to the number of possible input values $2^{\chi n}$, the number of rounds must be at least $r \simeq n \cdot \log_3(2)$.

However, just reaching the full degree is not sufficient to prevent the interpolation attack. First, the polynomial should be dense to guarantee that most monomials occur in it. As showed in [32], the interpolation polynomial is dense when working in $\mathbb{F}_p$. The situation is instead different when working in $\mathbb{F}_{2^n}$, where one needs at least $1 + \lceil \log_3(2^n - 1) \rceil + \lceil \log_2(t) \rceil$ rounds in order to guarantee that E_k is *dense*.

Since S-Box$^{-1}(x) = x^{1/3} = x^{(2^{n+1}-1)/3}$ has an higher degree than S-Box$(x) = x^3$, we do not expect the attack performs better when considering the backward direction instead of the forward one.

Secondly, we consider the algebraic degree not at round r but at round $r - 1$ to account for partial S-Box case where the degree increase is delayed for $t - 1$ words by 1 round. As a result, the total number of rounds R must satisfy [17]

$$R \geq 1 + \lceil n \cdot \log_3(2) \rceil + \Phi(t)$$

to thwart the interpolation attack where

$$\Phi(t) = \begin{cases} \log_2(t) & \text{working in } \mathbb{F}_{2^n} \\ \log_3(t) & \text{working in } \mathbb{F}_p \end{cases}$$

Security up to $2^M \leq 2^N$. For completeness, we present the number of rounds necessary to provide security up to 2^M (that is, data and computational cost of the attacker upper bounded by 2^M).

Using the same argumentation given before, the number of rounds must satisfy

$$(3^{r-\Phi(t)-1} + 1)^{\chi} \approx 2^{\min\{M, n\cdot\chi\}}$$

that is $r \geq 1 + \Phi(t) + \min\{n, M/\chi\} \cdot \log_3(2)$. The maximum number of attacked rounds is achieved for $\chi = 1$. As a result, we have $R_P + R_F = \big(1 + \lceil \log_3(2) \cdot \min(M, n) \rceil\big) + \Phi(t)$.

Gröbner Basis Attack. We consider the Gröbner Basis Attack in the same setting: some permutation inputs are unknown and the rest are known to the

[17] We emphasize that in this analysis we do not take into account the cost to construct the interpolation polynomial, which is (in general) non-negligible.

attacker. Given some words of the permutation output, they have to find the unknowns.

For generic systems, the complexity of computing a Gröbner basis for a system of $\mathfrak{N}$ polynomials f_i in $\mathfrak{V}$ variables is $\mathcal{O}\left(\binom{\mathfrak{V}+D_{reg}}{D_{reg}}^{\omega}\right)$ operations over the base field $\mathbb{F}$ [15], where D_{reg} is the *degree of regularity* and $2 \leq \omega < 3$ is the linear algebra constant. We note that the memory requirement of these algorithms is of the same order as the running time. The degree of regularity depends on the degrees of the polynomials d and the number of polynomials $\mathfrak{N}$. When $\mathfrak{V} = \mathfrak{N}$, we have the simple closed form

$$D_{reg} := 1 + \sum_{i=0}^{\mathfrak{N}-1}(d_i - 1), \tag{23}$$

where d_i is the degree of the i-th polynomial f_i in the polynomial system we are trying to solve (see [4] for details). In the over-determined case, i.e., $\mathfrak{V} < \mathfrak{N}$, the degree of regularity can be estimated by developing the Hilbert series of an ideal generated by generic polynomials $\langle f_0, \ldots, f_{\mathfrak{N}-1}\rangle$ of degrees d_i (under the assumption that the polynomials behave like generic systems). Closed-form formulas for D_{reg} are known for some special cases, but not in general.

Full-permutation equations. In the first case we derive equations, one by word, for the entire r-round permutation. We consider the case when the number χ of unknown input variables equals the number of known output variables. Then we get χ equations of degree $D_3(r) = 3^r$ of χ variables, so the degree of regularity is

$$D_{reg} = 1 + \chi(3^r - 1) = 3^r - \chi + 1.$$

The attack complexity can be estimated by

$$\binom{\mathfrak{V} + D_{reg}}{D_{reg}}^2 \approx \binom{\chi 3^r}{\chi}^2 \approx \frac{(3^r)^{2\chi} e^{2\chi}}{6.3\chi}.$$

If we target a security level of M bits, the number of rounds to be attacked is calculated as

$$\frac{(3^r)^{2\chi} e^{2\chi}}{6.3\chi} \leq 2^{\min(M, n\chi)},$$

which implies $r \leq 2 + 0.32 \cdot \min(M/\chi, n)$. Note the maximum number of attacked rounds is achieved for $\chi = 1$.

Equations for each S-Box. Here we consider equations of degree 3 for each S-Box, which relate its inputs and outputs. Given χ unknown permutation inputs and χ known outputs, we get $(t-1)R_F + R_P + \chi$ unknown S-Boxes, and for each we use 1 variable (for its input). In total, we get $(t-1)R_F + R_P$ equations for the S-Box inputs in all rounds, and χ equations for the last-round outputs. Denoting $q = (t-1)R_F + R_P + \chi$, the degree of regularity is

$$D_{reg} = 1 + 2q.$$

The attack complexity can be estimated by

$$\binom{\mathfrak{V} + D_{reg}}{D_{reg}}^2 \approx \binom{3q}{q}^2 \approx 2^{5.4q}.$$

If we target a security level of M bits, the number of rounds to be attacked is calculated as

$$2^{5.4((t-1)R_F+R_P+\chi)} \leq 2^{\min(M,n\chi)}$$

Clearly the maximum number of rounds to be attacked is achieved for maximum χ, and we get $tR_F + R_P \leq 0.18M$.

Combining the two strategies together, we get the following conditions:

$$R_F + R_P \geq 2 + 0.32\min(M, n) \tag{24}$$

$$tR_F + R_P \geq 0.18M \tag{25}$$

Higher-Order Differential Attack. A well-known result from the theory of Boolean functions is that if the algebraic degree of a vectorial Boolean function $f(\cdot)$ (like a permutation) is d, then the sum over the outputs of the function applied to all elements of a vector space $\mathcal{V}$ of dimension $\geq d+1$ is zero (as is the sum of all inputs, i.e., the elements of the vector space). The same property holds for affine vector spaces of the form $\{v + c \,|\, v \in \mathcal{V}\}$ for arbitrary constant c

$$\bigoplus_{v \in \mathcal{V} \oplus c} v = \bigoplus_{v \in \mathcal{V} \oplus c} f(v) = 0.$$

This is the property exploited by higher-order differential attack [25].

Working at word level, the number of rounds (R_F+R_P) given by the interpolation attack provides security also against higher-order differential attacks. Indeed, for the interpolation attack it is required that the degree d after r rounds satisfies $d \geq 2^N$. Instead, for higher-order differentials (working at word level), it is sufficient that $d \geq N+1$. The conclusion follows immediately.

What happens if one works - instead - on a bit level? To prevent such attacks, ideally we would like to be able to make a statement such as "After r rounds there is no output bit and no input subspace of dimension d' s.t. the derivative of the polynomial representation of the output bit with respect to this subspace is the zero polynomial." To achieve such a goal, we need to estimate the *growth of the boolean degree*. First of all, the degree of the S-Box $f(x) = x^3$ in its algebraic representation in $\mathbb{F}_{2^n}$ is only 2. Thus, clearly the boolean degree of the permutation after r rounds is bounded from above by 2^r. It is furthermore generally bounded from above by $N-1$ as it is a permutation.

However, it turns out that the boolean degree grows slower than expected because the monomial x^{2^k} is a linear transformation in $\mathbb{F}_2^n$, and a high degree in $\mathbb{F}_{2^n}$ may not imply a high degree in $\mathbb{F}_2^n$. Nevertheless we assume that the boolean degree of f is at least q if f over $\mathbb{F}_{2^n}$ contains a monomial x^d where d has Hamming weight q. From the interpolation attack details we know that after $r+\log_2 t$ rounds the polynomial of f is dense and thus contains most of the monomials of degree 3^r and smaller. We now recall that for integer d there are at least $\log d$ integers smaller than d with Hamming weight $\lfloor \log_2 d \rfloor - 1$. Therefore for a polynomial in 1 variable, that is dense up to degree d, the boolean degree is at least $\lfloor \log_2 d \rfloor - 1$. For a polynomial in χ variables that is dense up to total degree d, we can find a monomial with degree up to d/χ in each variable, so the boolean degree would be $\chi(\lfloor \log_2(d/chi) \rfloor) - \chi$. Thus if bits in χ input words are unknown, the boolean degree after r rounds can be lower bounded as $\chi(\lfloor \log_2(3^{r-\log_2 t}/chi) \rfloor) - \chi$. As long as the degree is smaller than $\min\{M, \chi n\}$, we get a valid attack. Therefore we have the condition for the number of attacked rounds:

$$\log_2(3^{r-\log_2 t}/chi) - 2 \leq \min\{M/\chi, n\}.$$

For $M < n$ the maximum number of rounds is reached for $\chi = 1$, whereas for $M \geq n$ the maximum is reached for $\chi = M/n$. Eventually we get that at most

$$r = 0.63 \min\{M, n\} + 2 + \log_2 t$$

rounds can be attacked.

Higher-Order Differential Attacks on $\mathbb{F}_p$. Here we emphasize an important difference between the higher-order differential attack on $\mathbb{F}_{2^n}$ and on $\mathbb{F}_p$. Given a function $f(\cdot)$ of degree d, the sum over the outputs of the function applied to all elements of a vector space $\mathcal{V}$ of dimension $\geq d+1$ is zero.

The crucial point here is that the previous result holds if $\mathcal{V}$ is a (sub)space, and not only a generic set of elements. While $\mathbb{F}_{2^m}$ is always a subspace of $\mathbb{F}_{2^n}$ for each $m \leq n$, the only subspaces of $\mathbb{F}_p$ are $\{0\}$ and $\mathbb{F}_p$. It follows that the biggest subspace of $(\mathbb{F}_p)^t$ has dimension t, with respect to the biggest subspace of $(\mathbb{F}_{2^n})^t$, which has dimension $n \cdot t = N$.

As a result, in the case in which a permutation is instantiated over $\mathbb{F}_p$, a lower degree (and hence a smaller number of rounds) is sufficient to protect it from the higher-order differential attack with respect to the number of rounds for the $\mathbb{F}_{2^n}$ case. In more details, the number of rounds necessary to protect our design against the interpolation attack are sufficient in order to guarantee security against this attack also.

Zero-Sum Distinguishers. The fact that some inner primitive in a hash function has a relatively low degree can often be used to construct higher-order diff. distinguishers, or *zero-sum structures*. This direction has been investigated e.g. in [10] for two SHA-3 candidates, Luffa and Keccak. More generally, a zero-sum structure for a function $f(\cdot)$ is defined as a set Z of inputs z_i that sum to zero, and for which the corresponding outputs $f(z_i)$ also sum to zero, i.e. $\bigoplus_i z_i = \bigoplus_i f(z_i) = 0$. For an iterated function, the existence of zero sums is usually due either to the particular structure of the round function or to a low degree. Since it is expected that a randomly chosen function does not have many zero sums, the existence of several such sets can be seen as a distinguishing property of the internal function.

By using the *inside-out* technique, here we investigate the minimum number of rounds of $x^3 - \text{Poseidon}^\pi$ sufficient to prevent zero-sum structures.

Definition 1 (Zero-sum Partition [10]). *Let P be a permutation from $\mathbb{F}_{2^n}$ to $\mathbb{F}_{2^n}$. A zero-sum partition for P of size $K = 2^k \lneqq 2^n$ is a collection of 2^k disjoint sets $\{X_1, X_2, ..., X_k\}$ with the following properties:*

- $X_i = \{x^i_1, ..., x^i_{2^{n-k}}\} \subset \mathbb{F}_{2^n}$ *for each* $i = 1, ..., k$ *and* $\bigcup_{i=1}^{2^{n-k}} X_i = \mathbb{F}_{2^n}$,
- $\forall i = 1, ..., 2^k$: *the set* X_i *satisfies zero-sum* $\bigoplus_{j=1}^{2^k} x^i_j = \bigoplus_{j=1}^{2^k} P(x^i_j) = 0$.

We focus on creating zero-sum partitions of the permutation $P(\cdot)$ of the form $P(\cdot) = R_r \circ ... \circ R_1(\cdot)$, where all R_i are permutations over $\mathbb{F}_2^n$. Remember that for the permutation in a hash function, one can exploit any state starting from an intermediate state. Thus, assume one can find a set of texts $X = \{x_i\}_i$ and a set of texts $Y = \{y_i\}_i$ with the property $\bigoplus_i R_{r-1} \circ ... \circ R_{s+1}(y_i) = 0$ and $\bigoplus_i R_s \circ ... \circ R_1(x_i) = 0$ for a certain s. Working with the intermediate states (remember that there is no secret material), the idea is to choose texts in $X \bigoplus Y$: the inputs p_i are defined as the $(r-s)$-round decryptions of $X \bigoplus Y$, while the corresponding outputs c_i are defined as the s-round encryptions of $X \bigoplus Y$. This results into a zero-sum partition $\{p_i\}$ for the permutation P.

To avoid such an attack, we require that $R_{r-1} \circ ... \circ R_{s+1}(\cdot)$ and $R_s \circ ... \circ R_1(\cdot)$ have maximum degree. About the forward direction of the permutation, one can simply reuse the result already proposed for the higher-order differential discussed in the previous section, i.e. we need $0.63M + \log_2 t$ rounds to achieve the full boolean degree. In the other direction we limit to recall here that the algebraic degree of S-Box$(x) = x^{1/3}$ (i.e. the inverse S-Box) is $(n+1)/2$ (see Prop.1).

Details – Algebraic Degree of S-Box$^{-1}(x) = x^{1/3}$ in $GF(2^n)$

Proposition 1. *The algebraic degree of S-Box$^{-1}(x) = x^{1/3} = x^{(2^{n+1}-1)/3}$ is $(n+1)/2$ (remember that n is odd).*

Proof. We prove this result by induction.

For $n = 3$, it follows that S-Box$^{-1}(x) = x^{1/3} = x^5$. Since $x^5 = x^4 \cdot x$ and since x^4 is a linear operation in $GF(2^n)$, the result follows immediately.

Assume the result is true for $n - 1 = 2n' + 1$. Here we show that it works for $n = 2n' + 3$. Observe that

$$\frac{2^{n+1}-1}{3} = \frac{2^{2n'+4}-1}{3} = \frac{2^{2n'+4}-2^{2n'+2}}{3} + \frac{2^{2n'+2}-1}{3} = 2^{2n'+2} + \frac{2^{2n'+2}-1}{3}$$

thus

$$x^{\frac{2^{n+1}-1}{3}} = x^{2^{2n'+2}} \cdot x^{\frac{2^{2n'+2}-1}{3}}.$$

Since the exponent of the first term on the r.h.s. is a power of 2, it is linear in $GF(2^n)$. By the induction assumption, the second term has algebraic degree $(n-1)/2$. It follows that the algebraic degree is $(n+1)/2$. $\square$

C Security Analysis – x^5-Poseidon

For some practical applications, we need to work with a prime p s.t. $p = 1 \mod 3$. Since the cubic function x^3 is invertible if and only if $p = 2 \mod 3$, we need to change the S-Box for this particular case.

We decided to work with S-Box$(x) = x^5$ which is invertible if and only if $p \neq 1 \mod 3$. Since the analysis for this case is similar to the one just given for the cubic case, we limit ourselves here to briefly discuss the number of rounds necessary to guarantee security.

In the following, we limit ourselves to work only in GF(p) (due to our target application).

C.1 Statistical Attacks

Differential Cryptanalysis. As before, HadesFifth$^\pi$ instantiated by S-Box$(x) = x^5$ is secure against statistical attacks if and only if

$$R_F^{stat} \geq 6.$$

The main difference here is due to differential and linear attacks. In particular, since[18] $DP_{max}(\text{S-Box}(x) = x^5) = 4/p$ (or equivalently 2^{-n+2} in $\mathbb{F}_{2^n}$), it follows

[18] Note that

$$(x + \Delta_I)^5 - x^5 = \Delta_O$$

is an equation of degree 4, hence there are at most 4 different solutions.

that the minimum number of rounds necessary to guarantee security against linear and differential attacks is given by

$$R_F = \begin{cases} 4 & \text{if } 2t + 4 < N + 2 \cdot \lceil \log_2(p) \rceil - M \\ 6 & \text{if } 2t + 4 \geq N + 2 \cdot \lceil \log_2(p) \rceil - M \end{cases}$$

for a security level up to $2^M \leq 2^N$ (that is, in the case in which the data and the computational cost of the attacker is upper bounded by 2^M).

Linear Cryptanalysis. Similar considerations hold for linear cryptanalysis.

Rebound Attacks. Due to the same argumentation in order to provide security of HadesCubic instantiated by S-Box$(x) = x^3$ against the rebound attack, 6 rounds provide security also HadesFifth instantiated by S-Box$(x) = x^5$ against the rebound attack.

C.2 Algebraic Attacks

Interpolation Attack. Due to the previous analysis, the number of rounds necessary to prevent the interpolation attack is given by

$$R_F + R_P \geq R^{inter}(N, t) \equiv 1 + \left\lceil \log_5(2) \cdot \min\left\{ \lceil \log_2(p) \rceil ; M \right\} \right\rceil + \lceil \log_5(t) \rceil$$

working in $\mathbb{F}_p$. In particular, note that the degree of the encrypted function after r rounds is well approximated by 5^{r-1}, and where $\log_5(t)$ more rounds are necessary to guarantee that the polynomial is sparse. Since the degree of S-Box$^{-1}(x) = x^{1/5}$ is much higher:

$$\frac{1}{5} \mod p \equiv \begin{cases} \frac{4 \cdot p - 3}{5} & \text{if } p \mod 5 = 2 \\ \frac{2 \cdot p - 1}{5} & \text{if } p \mod 5 = 3 \\ \frac{3 \cdot p - 2}{5} & \text{if } p \mod 5 = 4 \end{cases}$$

the same number of rounds guarantee security in the case in which the attacker is performed in the decryption direction.

Higher-Order Diff. Attack. Since we present HadesFifth-Hash instantiated by S-Box$(x) = x^5$ just over GF(p), we refer to previous discussion against higher-order diff. attacks over $\mathbb{F}_p$, and we limit ourselves to remember that the number of rounds necessary to guarantee security against the interpolation attack is also also sufficient to guarantee security against higher-order diff. attacks.

Gröbner Basis Attack. Using exactly the same analysis as for x^3-Poseidon, we get the following conditions:

$$R_F + R_P \geq 2 + 0.21 \min(M, n) \tag{26}$$

$$tR_F + R_P \geq 0.14M \tag{27}$$

D Security Analysis – x^{-1}-Poseidon

Here we propose the security analysis of Poseidon instantiated[19] with $\text{S-Box}(x) = x^{-1}$ in $GF(p)$. In particular, we focus only on the attacks that *depend* on the details of the S-Box, like differential/linear attacks and the algebraic attacks.

D.1 Statistical Attacks

Since statistical attacks work in the same way in $GF(p)$ and $GF(2^n)$, in this subsection we do not distinguish the two cases.

Differential Attack. For simplicity[20], assume $N \geq 10$, that is, $n \geq 4$. Then, as for Poseidon with $\text{S-Box}(x) = x^3$, 8 rounds with full S-Box layers are largely sufficient to prevent differential and linear attacks. In particular, w.r.t. the cubic function, the inverse function $\text{S-Box}(x) = x^{-1}$ in $GF(2^n)$ is not APN but differentially 4-uniform [33]. This means that its differential probability is bounded by 2^{-n+2}.

Thus, consider the "weaker" permutation $R^3 \circ L \circ R^3(\cdot)$ as defined in (22), and focus on the "middle" 5 rounds. Since M is an MDS matrix, at least $2 \cdot (t+1)+1$ S-Boxes are active in the middle 5 rounds of the middle permutation. As a result, each characteristic has probability

$$(2^{-n+2})^{2\cdot(t+1)+1} = 2^{-N} \cdot 2^{-N-3n+4t+6} < 2^{-N}$$

since $[N + 3n = n \cdot (t+3)] > [4t+6]$. For $n \geq 4$, the result follows immediately.

Again, 4 rounds are sufficient if $n > t+2$. Indeed, each characteristic has probability

$$(2^{-n+2})^{t+2} = 2^{-N} \cdot 2^{-2n+2t+4} < 2^{-N}$$

since $n > t+2$.

Security up to $2^M \leq 2^N$. For completeness, we present the number of rounds necessary to provide security up to 2^M (that is, data and computational cost of the attacker upper bounded by 2^M). Using the same analysis as before, it turns out that

$$R_F = \begin{cases} 4 & \text{if } 2t+4 < N+2n-M \\ 6 & \text{if } 2t+4 \geq N+2n-M \end{cases}$$

Linear Cryptanalysis. Similar considerations hold for linear cryptanalysis.

[19] We do not have any practical application of HadesInverse$^\pi$ instantiated in $GF(2^n)$. For this reason, here we limit ourselves to consider the case $GF(p)$.

[20] Due to the MDS assumption, a $t \times t$ MDS matrix with elements in $GF(2^n)$ exists if $2t+1 \geq 2^n$. If $n=3$, then $t \leq 3$ which implies $N \leq 9$.

Rebound Attacks. Due to the same argumentation in order to provide security of HadesCubic instantiated by S-Box$(x) = x^3$ against the rebound attack, 6 rounds provide security also HadesInverse instantiated by S-Box$(x) = x^{-1}$ against the rebound attack.

D.2 Algebraic Attacks

Higher-Order Diff. Attack. We refer to previous discussion against higher-order diff. attacks over $\mathbb{F}_p$, and we limit ourselves to remember that the number of rounds necessary to guarantee security against the interpolation attack is also also sufficient to guarantee security against higher-order diff. attacks.

Interpolation Attack. As we have already seen, in an interpolation attack [23], the goal is to determine the polynomial representation of a state word. Since the inverse function has high degree, one may think that the interpolation attack can cover only few rounds in this case. However, exploiting the original idea proposed by Jakobsen and Knudsen in [23], it is possible to show that the following:

- for a full S-Box layer, the S-Box $f(x) = x^{-1}$ has the same behavior as the one of a function of algebraic degree t (i.e., the number of words)[21] "from the point of view" of the interpolation attack;
- for a partial S-Box layer (with a single S-Box), the S-Box $f(x) = x^{-1}$ has the same behavior as the one of a function of algebraic degree 2 "from the point of view" of the interpolation attack.

Note that the two previous cases lead to two completely different results, while we emphasize that the two previous cases (full or partial S-Box layer) are equivalent for a cubic S-Box. It follows that the choice to use partial or full S-Box layer in order to protect from algebraic attacks also depend on the details of the S-Box.

Full S-Box Layer. Firstly, consider $t = 1$. In this case, every encryption function can be written as

$$f(x) = \frac{x + A}{B \cdot x + C}$$

for *any* number of rounds and for some constants A, B, C. This means that 4 texts are sufficient to break the permutation.

Consider the case $t = 2$. Let $f_i^r(\cdot) \equiv \frac{Nf_i^r(\cdot)}{Df_i^r(\cdot)}$ (for $i = 0, 1$) be the interpolation polynomial at round r of the i-th word. By simple computation, the i-th word

[21] More precisely, the degree of S-Box$(x) = x^{-1} \equiv x^{p-2}$ "from the point of view of the interpolation attack" is $\min\{t, p-2\}$, where t is due to the fraction representation and $p-2$ is due to the "normal" representation. Since $2t \leq p+1$ in order to guarantee that a $t \times t$ MDS matrix with coefficients in $\mathbb{F}_p$ exists, it follows that $\min\{t, p-2\} = t$.

of the function at round $r+1$ (assuming a full S-Box layer) for $i = 0, 1$ can be written as

$$\begin{aligned}
f_i^{r+1}(x \equiv [x_0, x_1]) &= \frac{A}{f_0^r(x \equiv [x_0, x_1]) + k_0} + \frac{B}{f_1^r(x \equiv [x_0, x_1]) + k_1} = \\
&= \frac{A \cdot Df_0^r(x)}{Nf_0^r(x) + k_0 \cdot Df_0^r(x)} + \frac{B \cdot Df_1^r(x)}{Nf_1^r(x) + k_1 \cdot Df_1^r(x)} = \\
&= \frac{A \cdot \big[Nf_1^r(x) + k_1 \cdot Df_1^r(x)\big] \times Df_0^r(x) + B \cdot \big[Nf_0^r(x) + k_0 \cdot Df_0^r(x)\big] \times Df_1^r(x)}{\big[Nf_0^r(x) + k_0 \cdot Df_0^r(x)\big] \times \big[Nf_1^r(x) + k_1 \cdot Df_1^r(x)\big]} = \\
&= \frac{Nf_i^{r+1}(x \equiv [x_0, x_1])}{Df_i^{r+1}(x \equiv [x_0, x_1])}
\end{aligned}$$

for some constants A, B. It follows that the degree of the function increases at most by a factor of 2 (where the degree after the first round is 1). As a result, the number of unknown coefficients after r rounds is at most $2 \cdot (2^{r-1} + 1)^2$, where the degree of the numerator (and so the number of unknown coefficients) is always less or equal than the degree of the denominator.

As a result, the number of unknown coefficients after r rounds for t words is approximately

$$2 \cdot (t^{r-1} + 1)^t.$$

The permutation can be considered secure if $2 \cdot (t^{r-1} + 1)^t \simeq 2^N$, that is, $t^{r-1} \simeq p$, which implies

$$r \geq \log_t(2) \cdot \log_2(p) + 1.$$

As a result, the total number of rounds (with full S-Box layer) must be

$$R_F \geq \log_t(2) \cdot n + 2 = 2 + \frac{\log_2(p)}{\log_2(t)}.$$

Partial S-Box Layer. Referring to the expression of f_i^r given before, it possible to note that all denominators at rounds r (for any r) are in general equal, while all numerators are in general different, that is

$$\forall i, j \in [0, 1, ..., t-1] : \qquad Df_i^r = Df_j^r.$$

This observation seems to have no effect on the complexity of the previous attack. Indeed, since the S-Box are applied at each word and since the numerators are different, it turns out that the denominators of S-Box(f^r) (which correspond to the numerator of f^r) are all different.

However, this has an important effect in the case in which we work with a partial non-linear layer, e.g. a non-linear layer composed of a single S-Box. Consider first the case $t = 2$ assuming the S-Box is applied only on the first word (we use the same notation as before):

$$\begin{aligned}
f_i^{r+1}(x \equiv [x_0, x_1]) &= \frac{A}{f_0^r(x \equiv [x_0, x_1]) + k_0} + B \cdot \big[f_1^r(x \equiv [x_0, x_1]) + k_1\big] = \\
&= \frac{A \cdot Df_0^r(x)}{Nf_0^r(x) + k_0 \cdot Df_0^r(x)} + \frac{B \cdot \big[Nf_1^r(x) + k_1 \cdot Df_1^r(x)\big]}{Df_1^r(x)} = \\
&= \frac{A \cdot Df_0^r(x) \times Df_1^r(x) + B \cdot \big[Nf_1^r(x) + k_1 \cdot Df_1^r(x)\big] \times \big[Nf_0^r(x) + k_0 \cdot Df_0^r(x)\big]}{\big[Nf_0^r(x) + k_0 \cdot Df_0^r(x)\big] \times Df_1^r(x)} = \\
&= \frac{Nf_i^{r+1}(x \equiv [x_0, x_1])}{Df_i^{r+1}(x \equiv [x_0, x_1])}
\end{aligned}$$

In this case, there is no difference w.r.t. the previous case.

Consider now the case $t \geq 3$. By previous observation, it follows that $Df_i^r(x) = Df_j^r(x)$ for each $i, j \geq 1$, which implies that

$$Df_i^{r+1} = \big[Nf_0^r(x) + k_0 \cdot Df_0^r(x)\big] \times Df_1^r(x)$$

also for the case $t \geq 3$. This fact has a huge impact on the number of monomials of the corresponding polynomial at round r. Indeed, the number of unknown coefficients after r rounds for t words is approximately

$$2 \cdot (2^{r-1} + 1)^t,$$

which is much smaller than $2 \cdot (t^{r-1} + 1)^t$ for large t. The permutation can be considered secure if $2 \cdot (2^{r-1} + 1)^t \simeq 2^N$, that is, $2^{r-1} \simeq p$, which implies

$$r \geq \log_2(p) + 1.$$

As a result, the total number of rounds (with full S-Box layer) must be

$$R \equiv R_P + R_F \geq \log_2(p) + 1.$$

Actually, the previous result can be improved. Since at least $R_F \geq 6$ rounds have a full S-Box layer, it follows that the number of unknown coefficients after $R = R_P + R_F$ rounds for t words is approximately

$$2 \cdot (2^{R_P} \cdot t^{R_F - 1} + 1)^t \equiv 2 \cdot (2^{R_P + (R_F - 1) \cdot \log_2(t)} + 1)^t.$$

The permutation can be considered secure if $2 \cdot (2^{R_P + (R_F - 1) \cdot \log_2(t)} + 1)^t \simeq 2^N$, that is

$$R_P + (R_F - 1) \cdot \log_2(t) \geq \log_2(p) + 1.$$

As a result, the total number of rounds (with full S-Box layer) must be

$$R_P + \log_2(t) \cdot R_F \geq R^{inter}(N, t) \geq 2 + \log_2(p) + \log_2(t).$$

Security up to $2^M \leq 2^N - 1$ S-Box Layer. For completeness, we present the number of rounds necessary to provide security up to 2^M (that is, data and computational cost of the attacker upper bounded by 2^M).

Using the same argumentation given before, the number of rounds must satisfy

$$2 \cdot (2^{R_P + (R_F - 1) \cdot \log_2(t)} + 1)^\chi \approx 2^{\min\{M, \log_2(p) \cdot \chi\}},$$

that is

$$R_P + R_F \cdot \log_2(t) \geq R^{inter}(N, t, M) = 2 + \log_2(t) + \min\{M, \log_2(p)\}$$

where the maximum number of attacked rounds is achieved for $\chi = 1$.

D.3 Gröbner Basis

We use the same setting with χ unknown inputs and χ known outputs.

After $r \geq 1$ rounds and using the "fraction representation" just proposed for the interpolation attack, the minimum degree of a variable in the output polynomials is t^{r-1}, using the equivalence

$$f(x) \equiv \frac{Nf(x)}{Df(x)} = C \quad \text{if and only if} \quad Nf(x) = C \cdot Df(x).$$

Therefore we get χ equations of degree t^{r-1} of χ variables, so the degree of regularity is

$$D_{reg} = 1 + \chi(t^{r-1} - 1).$$

If we target the security level of M bits, the number of rounds to be attacked is calculated as

$$\frac{(t^{r-1})^{2\chi} e^{2\chi}}{6.3\chi} \leq 2^{\min(M, n\chi)}$$

which implies $r \leq 2 + 0.5 \log_2 t \cdot \min(M/\chi, n)$. Note the maximum number of attacked rounds is achieved for $\chi = 1$.

Equations for each S-box. Here we consider equations of degree 2 for each S-box, which relate its inputs and outputs. Given χ unknown permutation inputs and χ known outputs, we get $(t-1)R_F + R_P + \chi$ unknown S-boxes, and for each we use 1 variable – for its input. In total we get $(t-1)R_F + R_P$ equations for the S-box inputs in all rounds, and χ equations for the last round outputs. Denoting $q = (t-1)R_F + R_P + \chi$, the degree of regularity is

$$D_{reg} = 1 + q.$$

The attack complexity is lower bounded by

$$\binom{\mathfrak{V} + D_{reg}}{D_{reg}}^2 \approx \binom{2q}{q}^2 \approx 2^{3.92q}.$$

If we target the security level of M bits, the number of rounds to be attacked is calculated as

$$2^{3.92((t-1)R_F+R_P+\chi)} \leq 2^{\min(M,n\chi)}$$

Clearly the maximum number of rounds to be attacked is achieved for maximum χ, and we get $tR_F + R_P \leq 0.25M$.