UNIVERSITÄT KLAGENFURT

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND CYBERSECURITY

UNIVERSITÀ DEGLI STUDI DI UDINE
hic sunt futura

DEPARTMENT OF MATHEMATICS, COMPUTER SCIENCE AND PHYSICS

Master thesis for Double-Degree in
Cybersecurity and Artificial Intelligence

Affine Cover Gröbner Basis Cryptanalysis of Lookup-Table AO Hash Functions

Author
Filippo Merlo

Supervisor
Prof. Pietro Di Gianantonio

Co-Supervisors
Prof. Arnab Roy
Prof. Elisabeth Oswald

2025-2026

Institute Contacts
Università degli Studi di Udine: Dipartimento di Scienze Matematiche, Informatiche e Fisiche
Via delle Scienze, 206, 33100 Udine (IT)
+39 0432 558400
info.dmif@uniud.it
`https://www.dmif.uniud.it/`

Author Contacts
Filippo Merlo
merlo.filippo@spes.uniud.it

© 2026 Filippo Merlo
This work is shared under the Creative Commons 4.0 License Attribution-NonCommercial-ShareAlike.

Ai miei cari nonni.

Acknowledgements

Ringrazio profondamente mia mamma Beatrice, mio papà Luigi, mio fratello Giacomo e i miei fratellini pelosi per avermi dato l'amore e il supporto che mi hanno permesso di vivere sempre in modo sereno e sorridente. Ringrazio anche le mie zie e i miei zii per avermi sempre accolto come in una seconda casa, senza farmi mai mancare nulla e per avermi trattato al pari di un loro figlio. Sono orgoglioso di tutta questa grande famiglia, di come mi hanno cresciuto e dei valori che mi hanno trasmesso.

Un grazie di cuore ad Arianna, per avermi dato la possibilità di vivere buona parte della mia vita con lei, per avermi dato modo di crescere come persona, riempiendo i miei vuoti e smorzando i miei difetti con la sua comprensione. La ringrazio per essere stata sempre un ottimo esempio da seguire, sia per l'impegno sia per l'intelligenza, e un luogo sicuro in cui tornare.

Infine, ci tengo a ringraziare chi mi ha ispirato e motivato, aiutandomi inconsapevolmente a essere chi sono ora, in particolar modo mio zio Bopo, senza il quale questo percorso non sarebbe mai iniziato.

Abstract

Arithmetization-Oriented (AO) hash functions are designed to operate efficiently inside Zero-Knowledge proof systems. Among them, the recently proposed *Skyscraper* operates over large prime fields $\mathbb{F}_p$ ($p \geq 2^{248}$) and combines a squaring operation with a lookup table defined over a binary extension field $\mathbb{F}_{2^\omega}$. This switch between fields complicates Gröbner basis cryptanalysis, since representing the lookup table over $\mathbb{F}_p$ requires an interpolation polynomial of high degree, which makes the resulting polynomial system dense.

This thesis replaces the interpolation polynomial with a few *affine agreement pieces* $L_i : \mathbb{F}_p \to \mathbb{F}_p$, each of which agrees with the lookup table *exactly* on a subset of its domain, with values constrained into $\mathbb{F}_{2^\omega}$. Enumerating which piece applies to each chunk, solving the resulting low-degree system with a Gröbner basis algorithm and verifying every candidate against the permutation yields a hybrid attack, whose cost depends mainly on how quickly a correct combination of pieces is found. We provide a combinatorial model of how often each combination agrees with the lookup table, accounting for the coupling between chunks introduced by the field boundary (valid inputs are the canonical representatives $0 \leq x < p$), and we use it to derive ranking criteria for the enumeration together with bounds on the search cost. The evaluated implementation applies only a structural restriction on the most significant chunk of each lookup layer, but are proposed as improvements, an enumeration sorting by the model's ranking and a reachability filter.

On the parameter-reduced instances tested, the attack is faster than the timings reported by the designers, even though they were measured on different hardware. The attack improves by factors ranging from about two times to almost three orders of magnitude and also solves parameter settings that were not reported. The analysis also shows that some combinations of affine pieces occur more often than others, caused by a clustering phenomenon that depends on the prime p. However, this hybrid attack does not threaten Skyscraper at its recommended parameters, but it shows that the representation of a lookup table has a measurable impact on the cost of Gröbner basis cryptanalysis.

Keywords: Zero-Knowledge , Arithmetization-Oriented hash functions , Skyscraper , lookup tables , algebraic cryptanalysis , Gröbner basis, affine cover , probabilistic attacks

Contents

Chapter 1 Introduction — **Page 1**

1.1 AO Designs 1
1.2 Related Work 2
1.3 Contributions 3
1.4 Outline 4

Chapter 2 Mathematical Background — **Page 5**

2.1 Finite Fields 5
2.2 Ideals, Varieties and Gröbner Bases 6
- Monomial Orderings 8
- Shape Position 9
- Gröbner Basis Complexity 10

2.3 CICO Problem 11

Chapter 3 Skyscraper — **Page 13**

3.1 Skyscraper Hash Function 13
- State and Sponge Construction 13
- Permutation $\mathcal{P}$ 14

3.2 Skyscraper-CICO 18
3.3 Polynomial Model 18
- Gröbner Basis Complexity 20

Chapter 4 Affine Cover Attack — **Page 23**

4.1 Piecewise Affine Approximation 23
4.2 Cover Selection Heuristics 25
4.3 Cost Model 26
4.4 Reduced Polynomial System 27
4.5 Hybrid Attack 28
4.6 B' Structure Exploitation 29

Chapter 5 Probabilistic Analysis Page 31
5.1 Bar Layer Approximation 31
- Field-Switch 32
- Chaining Bar Layers 34
5.2 Trail Selection 35
- Clustering Phenomenon 36
5.3 Parameter Dependence 37
5.4 Scaling to Real Parameters 38
5.5 Attack Complexity 39
5.6 Cross-Layer Heuristics 39
5.7 Expected Search Cost 41

Chapter 6 Experimental Results Page 43
6.1 Experimental Setup 43
6.2 Average Running Time 44
6.3 Combination Ratio 46

Chapter 7 Conclusion Page 49
7.1 Summary 49
7.2 Limitations 50
7.3 Future Work 50
- Attack Improvements 50
- Extension to Other Primitives 51

ChapterA Implementations Page 53
A.1 Affine Cover 53
A.2 Enumeration Order 54
A.3 Attack Loop 54

Chapter B Affine Covers Page 57

Chapter Bibliography Page 59

List of Figures

3.1 Classical absorb-and-squeeze sponge construction. 14
3.2 Skyscraper's permutation function $\mathcal{P}$ for $R = 10$ rounds. 14
3.3 SQUARE function S. 15
3.4 BAR function B. 16
3.5 B' function over m chunks, each of ω bits. 17

4.1 Scatter plot of $(x, T(x))$ for different ω. 24
4.2 Affine cover of $(x, T(x))$ for different ω. 25
4.3 B' function compression for primes for $m = 4$. 30

5.1 Full-round Skyscraper cross-layer propagation. 40

6.1 Histogram of combination-ratio outcomes underlying Table 6.4. 46

List of Tables

3.1 The $\omega = 4$ lookup table T. 15
3.2 Authors results of Gröbner basis cryptanalysis against CICO Skyscraper. 17
3.3 Lagrange interpolant of the $\omega = 4$ lookup table. 21
3.4 Reduced Gröbner basis sizes for a small B'-only toy instance. 21

5.1 Trails ranking by agreement rate for $p = 587$. 36
5.2 Agreement rate dependence on the prime structure. 37

6.1 Hardware and Software specifications of the test environment. 43
6.2 Skyscraper-CICO pre-image search solving times for $\omega = 4$. 44
6.3 Speedup of the hybrid attack over the baseline. 45
6.4 Combinations ratio of Skyscraper-CICO probabilistic attack results. 47

B.1 Affine cover for $\omega = 4$. 57
B.2 Affine cover for $\omega = 8$. 58

List of Listings

A.1 Affine cover definition. 53
A.2 Lookup-table T function. 54
A.3 Enumeration of Bar-layer cases. 54
A.4 Hybrid attack loop. 55
A.5 Reduced system Gröbner basis computation. 55

1 Introduction

Zero-knowledge (ZK) proof systems allow a prover to convince a verifier that a statement is true without revealing any information beyond its validity. Over the last decade, ZK proofs have moved from a purely theoretical tool to a practical building block for blockchain protocols, anonymous authentication and verifiable computation. In these settings, a prover must repeatedly demonstrate the correct execution of a computation, such as a hash evaluation or a state transition, without exposing the underlying data. These proof systems also showed that primitives designed for classical settings are poorly suited to ZK scenarios, which led to the development of new cryptographic designs for ZK implementations.

1.1 AO Designs

Most ZK proof systems (*e.g.*, SNARKs and STARKs) express the statement to be proven as an arithmetic circuit over a large prime field $\mathbb{F}_p$, or, equivalently, as a system of polynomial constraints that the prover's witness must satisfy. The cost of generating a proof is dominated by the number of multiplication gates of this circuit, rather than by the number of bit operations a CPU would need to evaluate the same function. Traditional symmetric primitives, such as AES or SHA, are designed for the opposite cost model. They are efficient on real hardware because they rely on bitwise operations (*e.g.*, XOR, bit rotations and table lookups) that map cheaply onto CPU instructions, but they translate poorly into arithmetic circuits. Using AES or SHA inside a ZK statement therefore makes the circuit grow substantially and the cost of proof generation becomes several orders of magnitude higher than with a primitive designed for the arithmetic cost model.

This observation motivated the design of *Arithmetization-Oriented* (AO) symmetric primitives, *i.e.*, block ciphers and hash functions whose internal operations are chosen to minimize the number of multiplication gates and constraints when expressed over $\mathbb{F}_p$, rather than the number of CPU cycles. Popular examples include MiMC [1], GMiMC [2], Poseidon [3], Rescue-Prime [4], Griffin [5], `ReinforcedConcrete` [6], `Monolith` [7] and, more recently, Skyscraper [8]. As for any class of cryptographic primitive, the security of these designs is assessed against the attacks that are native to their algebraic structure. For AO primitives, *algebraic cryptanalysis* via Gröbner basis computation is one of the most effective attacks and is the one considered in this thesis. The algebraic security of AO primitives has been analyzed both by their designers and by independent third parties, typically through direct Gröbner basis computation on the CICO problem or through dedicated attacks that exploit the specific shape of the round function.

Early AO designs such as MiMC, Poseidon and Rescue-Prime build their nonlinear layer

directly on a power map $x \mapsto x^d$ over $\mathbb{F}_p$, for a small exponent d coprime to $p-1$, which keeps every round expressible as a low-degree polynomial. This makes the nonlinear layer efficient in circuits, but it also reduces the security of these constructions since Gröbner basis algorithms perform well [9] on single power maps. This is one of the reasons why new versions (*e.g.*, POSEIDON2) and new design strategies have been proposed.

A more recent strategy, adopted by `ReinforcedConcrete`, `Monolith` and Skyscraper, is the *lookup-table approach*, where part of the round function operates not directly over $\mathbb{F}_p$ but over a binary extension field $\mathbb{F}_{2^\omega}$ of characteristic 2. This makes it possible to use a small nonlinear S-box $T : \mathbb{F}_{2^\omega} \to \mathbb{F}_{2^\omega}$ represented via a lookup table rather than a polynomial map. In such a construction, an element of $\mathbb{F}_p$ is decomposed into m chunks of ω bits each, every chunk is mapped through T independently and the resulting chunks are recomposed into a single element of $\mathbb{F}_p$. By alternating between a field of odd characteristic and a binary extension field, designers obtain strong nonlinearity and favorable differential and linear properties. At the same time, the arithmetic complexity of the rest of the permutation stays low. This thesis focuses on Skyscraper because its permutation structure and its lookup table are comparatively simple to model.

From the standpoint of algebraic cryptanalysis, this field switch is precisely what makes the lookup-table approach interesting to attack. Gröbner basis techniques are very powerful for solving polynomial systems, but their cost is governed by the *degree* and the *density* of the generators of the system. Representing the lookup table T as a polynomial over $\mathbb{F}_p$, for instance via Lagrange interpolation on $\{0, \ldots, 2^\omega - 1\}$, yields a polynomial of degree $2^\omega - 1$. This is the case for T, which was chosen for its S-box properties within $\mathbb{F}_{2^\omega}$ and not for its algebraic complexity within $\mathbb{F}_p$. Its polynomial is paired with some range constraints to assert that each chunk variable is a valid ω-bit integer, which are themselves polynomials of degree 2^ω, producing a system of dense high-degree equations. Such a system leads to a combinatorial explosion in the number of polynomial reductions performed by Gröbner basis algorithms, such as Buchberger's algorithm [10], F_4 [11] and F_5 [12].

1.2 Related Work

Algebraic attacks on AO primitives. As mentioned, first Gröbner basis attacks on AO primitives targeted designs based on power maps, such as MiMC and the MARVELlous family [13, 14], and these attack algorithms have been collected and presented in an intuitive way by Sauer and Szepieniec [15]. More recent works exploit the structure of the polynomial system in order to speed-up/avoid the Gröbner basis computation, like the algebraic FreeLunch approach [16], which has been extended beyond CICO-1 [17] and applied in resultant-based techniques [18]. Dedicated analyses also exist, including zero-dimensional Gröbner bases for POSEIDON [19] and Rescue [20], attacks based on subspace trails [9] and Gröbner basis attacks on Anemoi [21]. All these attacks were successful because exploited the algebraic structure of the primitives built on the low-degree power maps over $\mathbb{F}_p$, which lookup-table designs deliberately avoid. In fact, designs like `ReinforcedConcrete`, `Monolith` and Skyscraper that are based on lookup tables over a binary extension field, have been studied with different techniques.

Lookup-table primitives. Bak and Perrin studied the security of split-and-lookup-based ZK-friendly primitives in general [22] and specifically for `Monolith` and Skyscraper [23]. Have

been also proposed attacks against other designs like a collision attack for Tip4, Tip5 and `Monolith` [24], and a distinguisher on the full permutation of Skyscraper [25], which led the designers to propose a hardened 18-round variant. The common strategy for these attacks is to target the round structure of the permutation and/or the resulting systems.

Hybrid approaches. The idea of combining exhaustive guessing with Gröbner basis computation is known as the *hybrid approach* of Bettale, Faugère and Perret [26], in which a subset of the variables is guessed and the resulting, easier systems are solved. Our attack follows this paradigm, but it guesses which affine piece of the lookup table applies to each chunk, rather than the value of a variable, in order to replace the dense high-degree lookup table polynomial with the guessed linear one.

1.3 Contributions

This thesis addresses the following research question: *is the interpolation polynomial the only exploitable algebraic representation of a lookup-table S-box, or can the field-switching bottleneck be sidestepped by representing T differently?* To answer it, we propose the following contributions.

Exact case-split by affine cover. Rather than encoding the S-box T through its unique interpolation polynomial of degree $2^\omega - 1$, we observe that the graph of T, viewed as a finite set of points $(x, T(x))$ in $\mathbb{F}_p^2$, is covered *locally* by a small number of *affine* functions, each of which agrees exactly with T on a subset of its domain. Replacing T by one of these affine pieces, under the assumption that the input lies in the corresponding subset, lowers the degree of the equation from $2^\omega - 1$ to 1.

Hybrid enumerate-and-solve attack. Since the correct piece for a given input is not known in advance, we turn the case-split into an attack that enumerates the candidate cases across all the lookup-table applications. For each candidate, the attack solves the resulting low-degree system with a Gröbner basis algorithm, discards the cases where to is equal to the unit ideal, otherwise verifies the solution and returns it. We also analyze structural properties of the chunk rotation and recomposition of Skyscraper that could refine the enumeration order, although only the restriction of the most significant chunk is implemented in the evaluated attack.

Probabilistic analysis of the attack. We provide a combinatorial estimation of the probability that a given case agrees with the lookup table on a randomly chosen input, accounting for the coupling between chunks induced by the field boundary. We use this estimation to reason about which cases the attack should try first and to bound the expected cost of the search.

Experimental evaluation. We implement the attack and compare it with the Gröbner basis baseline reported in the Skyscraper design paper [8] on the same parameter-reduced instances. With respect to the designers' timings, we obtained speedups from about two times to almost three orders of magnitude, but we have also been able to solve parameter settings that were not reported in the original evaluation.

Note

Although Skyscraper is the case study of this thesis, the proposed technique applies more broadly to any lookup-table-based AO primitive.

Our result does *not* threaten the security of Skyscraper at its intended parameters ($p \geq 2^{248}$), since all the experiments were conducted on small parameter-reduced instances for reasons of computational feasibility. Its value instead lies in exposing a representation-level weakness in the way lookup-table S-boxes are presented to algebraic solvers and in quantifying its impact on the algebraic security of such designs. It also provides a general strategy that can be extended to other designs or improved to tackle full-scale parameters.

1.4 Outline

Chapter 2 reviews the mathematical background of the thesis, *i.e.*, finite fields and binary extension fields, the commutative algebra needed to define Gröbner basis and their computational complexity, and the generic CICO problem that models pre-image and collision search against a sponge-based hash function. Chapter 3 introduces the Skyscraper hash function, instantiates the CICO problem for it and derives its polynomial system. It then works through a concrete Gröbner basis computation to show where and why the field-switching bottleneck appears in practice. Chapter 4 presents the core contribution of the thesis, namely the exact case-split of the lookup table by affine cover, the resulting reduced polynomial system and the hybrid enumerate-and-solve attack built on it. It also includes the refinements derived from the chunk rotation of Skyscraper. Chapter 5 develops the probabilistic analysis of the attack, which explains the enumeration restriction applied in Chapter 4 and discusses how the attack scales beyond the parameter-reduced instances. Chapter 6 reports the experimental results against the parametrizations used in the analysis of the design paper and compares the two. Finally, Chapter 7 summarizes our findings, discusses the security implications and outlines the directions for future work. Lastly, the appendices contain the code of the implementation (Appendix A) and the affine covers used in the examples and experiments (Appendix B).

2 Mathematical Background

This chapter collects the mathematical tools used throughout the thesis. It reviews finite fields and, in particular, binary extension fields, since Skyscraper's lookup table is most naturally defined over one. Then it introduces the commutative-algebra vocabulary, thus ideals, varieties, monomial orderings and *Gröbner Basis* (GB), which are needed to define what a Gröbner basis computation is and what it returns. Finally, we define the generic *Constrained-Input/Constrained-Output* (CICO) problem, instantiated precisely for Skyscraper in Chapter 3, which is needed to construct formally the system of polynomial equations for pre-image and collision search against sponge-based hash functions.

2.1 Finite Fields

Definition: *Field*

A *field* is a set $\mathbb{K}$ equipped with two binary operations $+$ and $\cdot$ such that $(\mathbb{K}, +)$ is an Abelian group with identity 0, $(\mathbb{K} \setminus \{0\}, \cdot)$ is an Abelian group with identity 1, and $\cdot$ distributes over $+$. Every field has a *characteristic*: either 0, or the smallest prime p such that $\underbrace{1 + \cdots + 1}_{p} = 0$.

This thesis only concerns *finite* fields, all of which have prime characteristic.

Theorem: *Classification of finite fields*

For every prime p and every integer $n \geq 1$, there exists a finite field with exactly p^n elements, and any two finite fields with the same number of elements are isomorphic. We denote this field by $\mathbb{F}_{p^n}$.

Two special cases of finite fields are used throughout this thesis and their differences matter because Skyscraper's design deliberately exploits both:

Prime fields. When $n = 1$, $\mathbb{F}_p$ is simply the set $\{0, 1, \ldots, p-1\}$ with addition and multiplication performed modulo p. Such a field has odd characteristic and the Skyscraper permutation is primarily defined over it.

Binary extension fields. When $p = 2$ and $n > 1$, we have an even characteristic field $\mathbb{F}_{2^n}$. Contrarily to a prime field $\mathbb{F}_p$, operations cannot be simply performed modulo 2^n (for

example, 2 has no multiplicative inverse modulo 4), because the ring $\mathbb{Z}/2^n\mathbb{Z}$ is *not* a field for $n > 1$. Instead, $\mathbb{F}_{2^n}$ is constructed as a quotient of the polynomial ring $\mathbb{F}_2[x]$:

$$\mathbb{F}_{2^n} \cong \mathbb{F}_2[x]/(f(x)),$$

where $f(x) \in \mathbb{F}_2[x]$ is any *irreducible* polynomial of degree n. Concretely, an element of $\mathbb{F}_{2^n}$ is represented as a polynomial of degree less than n with coefficients in $\mathbb{F}_2$. Addition in $\mathbb{F}_{2^n}$ is simply a bitwise XOR, while multiplication is polynomial multiplication followed by reduction modulo $f(x)$. This is the reason binary extension fields are attractive as the domain of an S-box in a mixed-field design: addition is essentially free, while the field carries enough algebraic structure to define an invertible nonlinear map. In the case of Skyscraper, we will denote its field switch with $\mathbb{F}_{2^\omega}$ instead of $\mathbb{F}_{2^n}$.

It is the difference in how $\mathbb{F}_p$ and $\mathbb{F}_{2^\omega}$ are defined that produces the *field switch* problem introduced in Chapter 1. An element of $\mathbb{F}_p$ has no native interpretation as an element of $\mathbb{F}_{2^\omega}$, since their underlying sets, group structures and characteristics are all different. Passing a chunk of an $\mathbb{F}_p$-element through an $\mathbb{F}_{2^\omega}$-defined S-box, and later using the result inside an $\mathbb{F}_p$-arithmetic circuit, is therefore only possible by way of an explicit decomposition/recomposition step. In cryptanalysis this translates into an explicit polynomial encoding of the S-box over $\mathbb{F}_p$. In particular, a chunk z_j produced by decomposing $x \in \mathbb{F}_p$ is conceptually an element of $\mathbb{F}_{2^\omega}$. Once it is used as a variable in a polynomial system over $\mathbb{F}_p$ for Gröbner basis cryptanalysis, however, it is a value in the subset $\{0, \ldots, 2^\omega - 1\} \subset \mathbb{F}_p$ and not an element of the field $\mathbb{F}_{2^\omega}$ itself. The polynomial system reasons entirely within $\mathbb{F}_p$ and the S-box T defined over $\mathbb{F}_{2^\omega}$ enters only through its interpolation polynomial over $\mathbb{F}_p$, never through $\mathbb{F}_{2^\omega}$-native arithmetic.

2.2 Ideals, Varieties and Gröbner Bases

Let $\mathbb{K}$ be a field and let $\mathbb{K}[x_1, \ldots, x_n]$ denote the polynomial ring in n variables over $\mathbb{K}$.

Definition: *Ideal*

A subset $I \subseteq \mathbb{K}[x_1, \ldots, x_n]$ is an *ideal* if it satisfies:

(i) $0 \in I$;

(ii) if $f, g \in I$ then $f + g \in I$;

(iii) if $h \in \mathbb{K}[x_1, \ldots, x_n]$ and $f \in I$ then $h \cdot f \in I$.

Given a finite set $\{f_1, \ldots, f_s\} \subseteq \mathbb{K}[x_1, \ldots, x_n]$, the ideal generated by them is

$$\langle f_1, \ldots, f_s \rangle = \left\{ \sum_{i=1}^{s} h_i f_i \mid h_i \in \mathbb{K}[x_1, \ldots, x_n] \right\}.$$

Definition: *Affine variety*

Given the polynomials $f_1, \ldots, f_s \in \mathbb{K}[x_1, \ldots, x_n]$, the *affine variety* defined by them is

$$V(f_1, \ldots, f_s) = \left\{ (a_1, \ldots, a_n) \in \mathbb{K}^n \mid f_i(a_1, \ldots, a_n) = 0, \ \forall 1 \leq i \leq s \right\}.$$

For an ideal I, we denote by $V(I)$ the affine variety defined by I. Over an algebraically closed field $\overline{\mathbb{K}}$, Hilbert's Nullstellensatz states that $V(I) = \varnothing$ if and only if $1 \in I$. One direction is elementary and holds over any field: if $1 \in I$ then $V(I) = \varnothing$, so a Gröbner basis equal to $\{1\}$ certifies that the system has no solution. The converse is trivial, since $1 \notin I$ only guarantees a point over $\overline{\mathbb{K}}$, which does not need to lie in $\mathbb{K}$ nor correspond to a solution of the original problem. For this reason, every candidate solution extracted from a Gröbner basis is checked explicitly against the original problem.

Definition: *Zero-dimensional ideal*

An ideal $I \subseteq \mathbb{K}[x_1, \ldots, x_n]$ is *zero-dimensional* if its variety $V(I)$ (over the algebraic closure $\overline{\mathbb{K}}$) is a finite set of points.

A useful and standard sufficient condition for zero-dimensionality is the presence of the *field equations*, which is automatically satisfied when working with a finite field.

Proposition: *Field equations force zero-dimensionality*

Let $I \subseteq \mathbb{F}_p[x_1, \ldots, x_n]$ be an ideal containing $x_i^p - x_i$ for every $i = 1, \ldots, n$. Then $V(I)$, computed over $\overline{\mathbb{F}}_p$, is contained in $\mathbb{F}_p^n$; in particular it is finite and I is zero-dimensional.

Proof

Every element of $\mathbb{F}_p$ is a root of $x^p - x$ (Fermat's little theorem), which has degree p, hence $x^p - x = \prod_{a \in \mathbb{F}_p} (x - a)$. If $(a_1, \ldots, a_n) \in V(I)$ then $a_i^p - a_i = 0$ for every i, so each a_i is one of the roots above, *i.e.*, $a_i \in \mathbb{F}_p$. Therefore, $V(I) \subseteq \mathbb{F}_p^n$ is a set of p^n points.

The range constraints of Skyscraper, defined later in Section 3.3, are exactly a field-equation construction restricted to a subset, since they force each z_j into the finite set $\{0, \ldots, 2^\omega - 1\} \subset \mathbb{F}_p$ in the same way that $x_i^p - x_i$ forces x_i into all of $\mathbb{F}_p$. Consequently, the CICO systems of this thesis are zero-dimensional, since each chunk variable is confined to a finite set by its range constraint and all the remaining variables are determined by the decomposition/recomposition and SQUARE equations of the model.

Definition: *Degree of a zero-dimensional ideal*

For a zero-dimensional ideal I, the *degree* of I, written d_I, is the number of points of $V(I)$ over $\overline{\mathbb{K}}$, counted with multiplicity. Equivalently, $d_I = \dim_{\mathbb{K}}\big(\mathbb{K}[x_1, \ldots, x_n]/I\big)$, the dimension of the quotient ring as a $\mathbb{K}$-vector space.

Multiplicity is what distinguishes, *e.g.*, the ideal $\langle x - 6 \rangle$, which has degree 1 and describes a single simple point, from $\langle (x-6)^2 \rangle$, which has degree 2 and describes a single point counted twice. Indeed, the quotient ring $\mathbb{K}[x]/\langle (x-6)^2 \rangle$ is 2-dimensional over $\mathbb{K}$, even though both ideals have the same underlying variety $\{6\}$. This distinction is the reason why a Gröbner basis computation can, or cannot, hand us a solution directly without a separate root-finding step.

Definition: *Radical ideal*

An ideal I is *radical* if $f^k \in I$ for some $k \geq 1$ implies $f \in I$. Equivalently (for a zero-dimensional ideal), I is radical exactly when every point of $V(I)$ has multiplicity 1, *i.e.*, when d_I equals the number of *distinct* points of $V(I)$.

2.2.1 Monomial Orderings

Computing a Gröbner basis of $I = \langle f_1, \ldots, f_s \rangle$ provides structural information about the solution set $V(I)$. A key ingredient in the definition of a Gröbner basis is the choice of a *monomial ordering*, which imposes a total well-order on the monomials $x^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ (with $\alpha \in \mathbb{Z}_{\geq 0}^n$) consistent with multiplication and division.

Definition: *Monomial ordering*

A *monomial ordering* $\succ$ on $\mathbb{K}[x_1, \ldots, x_n]$ is a relation on $\mathbb{Z}_{\geq 0}^n$ satisfying:

(i) $\succ$ is a total ordering on $\mathbb{Z}_{\geq 0}^n$;

(ii) if $\alpha \succ \beta$ and $\gamma \in \mathbb{Z}_{\geq 0}^n$ then $\alpha + \gamma \succ \beta + \gamma$;

(iii) $\succ$ is a well-ordering: every non-empty subset of $\mathbb{Z}_{\geq 0}^n$ has a smallest element.

The most commonly used orderings are:

- **Lexicographic** (`Lex`): $\alpha \succ_{\text{Lex}} \beta$ if the leftmost nonzero entry of $\alpha - \beta$ is positive;
- **Graded lexicographic** (`DegLex`): first compares the total degree, *i.e.*, $|\alpha| = \sum_i \alpha_i$, and, if equal, falls back to lexicographic order;
- **Graded reverse lexicographic** (`DegRevLex`): first compares the total degree, and, if equal, breaks the tie by comparing the *rightmost* nonzero entry of $\alpha - \beta$, favoring the ordering in which that entry is *negative*. This is the most commonly used ordering in Gröbner basis algorithms as it usually leads to low-degree polynomials.

Given a monomial ordering $\succ$ and a nonzero polynomial $f = \sum_\alpha c_\alpha x^\alpha$, we use $\mathrm{LM}(f)$, $\mathrm{LC}(f)$, $\mathrm{LT}(f)$ to denote the leading monomial, the leading coefficient and the leading term of f with respect to $\succ$.

Definition: *Gröbner basis*

Fix a monomial ordering on $\mathbb{K}[x_1, \ldots, x_n]$. A finite subset $G = \{ g_1, \ldots, g_t \}$ of an ideal $I \subseteq \mathbb{K}[x_1, \ldots, x_n]$ with $I \neq \{ 0 \}$, is a *Gröbner basis* of I if

$$\langle \mathrm{LT}(g_1), \ldots, \mathrm{LT}(g_t) \rangle = \langle \mathrm{LT}(I) \rangle,$$

where $\langle \mathrm{LT}(I) \rangle$ denotes the ideal generated by all leading terms of the elements of I.

Every nonzero ideal admits a Gröbner basis and its finiteness follows from Hilbert's Basis Theorem, thus such a basis can be constructed via Buchberger's algorithm [10]. Moreover, a Gröbner basis is called *reduced* if every g_i is monic[1] and no monomial of g_i lies in $\langle \mathrm{LT}(G \setminus \{ g_i \}) \rangle$, and such a basis is unique with respect to the chosen monomial ordering.

[1] That is, its leading coefficient is equal to 1.

2.2.2 Shape Position

The practical relevance of Gröbner basis in cryptography lies in the fact that, for a zero-dimensional ideal, the reduced Gröbner basis with respect to a lexicographic order often exhibits a triangular structure from which the solutions can be obtained by back-substitution.

Definition: *Shape position*

A zero-dimensional radical ideal $I \subseteq \mathbb{K}[x_1, \ldots, x_n]$ is in *shape position* [27] (with respect to the variable ordering $x_1 \succ \cdots \succ x_n$) if its reduced Lex Gröbner basis has the form

$$G = \{\, x_1 - g_1(x_n),\ \ldots,\ x_{n-1} - g_{n-1}(x_n),\ g_n(x_n) \,\},$$

i.e., a single multivariate-free univariate polynomial $g_n(x_n)$ in the last variable, together with one polynomial per remaining variable expressing it directly in terms of x_n.

In shape position, once the roots of the univariate polynomial $g_n(x_n)$ are found, every other coordinate is recovered by a single evaluation $x_i = g_i(x_n)$, thus solving such system reduces to a univariate polynomial plus back-substitution.

However, as argued in Section 3.2, the reduced Gröbner basis of Skyscraper's CICO system does not merely reduce solving to a univariate polynomial. Instead, it returns the solution *immediately*, with every generator already of the form $x_i - a_i$ for a constant $a_i \in \mathbb{F}_p$, so that no root-finding step is needed at all. This is caused by a degenerate, degree-1 case of the shape position defined above.

Remark: *Degree-1 shape position*

Specializing the shape-position form above to the case where $g_n(x_n)$ has degree 1, *i.e.*, $g_n(x_n) = x_n - a_n$ for some constant $a_n \in \mathbb{K}$ collapses every other generator $g_i(x_n)$ to a constant a_i as well, since each g_i is obtained by reducing modulo a linear polynomial in x_n. Equivalently, if I is radical and its variety $V(I)$ is a single point $(a_1, \ldots, a_n) \in \mathbb{K}^n$, I reduced Gröbner basis is $\{\, x_1 - a_1, \ldots, x_n - a_n \,\}$, and this holds with respect to *any* monomial ordering, not only Lex. Indeed, the leading term of $x_i - a_i$ is x_i regardless of the ordering chosen, so this set already satisfies $\langle \mathrm{LT}(G) \rangle = \langle x_1, \ldots, x_n \rangle = \langle \mathrm{LT}(I) \rangle$ and is trivially reduced. By uniqueness of the reduced Gröbner basis, it is therefore the reduced Gröbner basis of I under every ordering.

This degenerate case defines precisely the conditions needed for a Gröbner basis to return the solution directly, without back-substitution or root-finding, *i.e.*, the ideal must be radical and the variety it defines must already be a single point. For a CICO instance with a unique solution over $\mathbb{F}_p$, both conditions are natural to expect since the system is zero-dimensional by construction and it is designed to isolate a single specific hash input or pre-image rather than a family of solutions. We stress that this behavior does not hold in the general case, *i.e.*, when a CICO instance has several solutions (*e.g.*, when searching for a collision rather than a specific pre-image), where instead the shape position defined above, at whatever degree the ideal actually has, must be considered.

2.2.3 Gröbner Basis Complexity

Buchberger's algorithm [10] computes a Gröbner basis of $I = \langle f_1, \ldots, f_s \rangle$ by repeatedly forming *S-polynomials* of pairs of generators (chosen to cancel their leading terms) and reducing them modulo the current generating set, adding any nonzero remainder to the set, until no S-polynomial reduces to something new. Each such reduction, and each generation of a new S-polynomial, is called a *critical pair reduction* and the overall running time of the algorithm is dominated by the total number of critical pairs considered and the cost of reducing each one.

In practice, Gröbner basis computations for cryptographic systems are not carried out with the original Buchberger algorithm, but with the F_4 [11] and F_5 [12] algorithms. These algorithms reformulate the same reduction process as sparse linear algebra over large matrices (rows are polynomials, columns are monomials), so that the reduction at once of many S-polynomials can be carried out via Gaussian elimination. This reformulation does not change what is being computed, but it changes the cost model, since the running time of F_4/F_5 is dominated by the cost of Gaussian elimination on matrices whose column count grows with the number of *distinct monomials* appearing during the computation. This number is in turn governed by the degree reached and the density of the polynomials involved.

The intermediate degree reached by a Gröbner basis computation is referred as solving degree.

Definition: *Solving degree*

The *solving degree* d_{solv} of a Gröbner basis computation for an ideal $I = \langle f_1, \ldots, f_s \rangle$ (with respect to a fixed monomial ordering and a fixed algorithm) is the maximum degree of any polynomial appearing at any point during the computation.

Because the solving degree depends on the algorithm and even, in principle, on implementation choices, it is generally hard to predict exactly for a given system [28]. It should not be confused with the degree of the ideal d_I, since d_{solv} measures how large the intermediate polynomials get *on the way* to computing a Gröbner basis and can be far larger than d_I even when d_I itself is small.

Definition: *Macaulay bound*

For a system of n polynomials in n variables of degrees $d_1, \ldots, d_n$ whose homogenizations form a regular sequence (*i.e.*, with no common zeros at infinity), the *Macaulay bound* is

$$d_{\text{M}} = \sum_{i=1}^{n} (d_i - 1) + 1 = d_1 + \cdots + d_n - n + 1.$$

It bounds the degree of regularity of the system and is commonly used as an (often loose) upper estimate of the solving degree of a Gröbner basis computation [28].

The systems considered in this thesis are overdetermined and do not satisfy this hypothesis in general, so we use the quantity only as a qualitative indicator of how the degrees of the generators drive the cost.

The Macaulay bound already explains, at a qualitative level, why the degree of the input generators matters so much. The bound grows with the sum of the input degrees, so a system built from degree-$(2^{\omega} - 1)$ polynomials admits, a priori, a far worse worst-case bound than

one built from degree-1 polynomials, even before accounting for how many such polynomials the system contains or how they interact. The Macaulay bound is typically far from tight for structured cryptographic systems but it correctly predicts the *direction* of the effect that the reduction in our proposed attack exploits, *i.e.*, lowering the degree of the generators lowers the ceiling on the degrees the computation can reach.

Degree alone does not determine cost, since a sparse polynomial of degree d like $x^d - 1$ is far cheaper to work with than a dense one of the same degree, *i.e.*, one containing a nonzero coefficient for every monomial up to degree d. In particular, the interpolation polynomial of a lookup table with no special structure is generically dense, because Lagrange interpolation produces a linear combination of the basis polynomials $\prod_{k \neq i}(x - x_k)$ and usually the resulting sum does not have cancelling terms. The field switch problem is not only caused by the high degree of the interpolating polynomial, but by the fact that it is also dense at that degree, resulting in a high Macaulay bound. Our solution, based on the affine cover of the lookup table function T, tackles both these aspects at once, since a linear function is trivially of minimal degree and minimal density.

2.3 CICO Problem

In Gröbner basis cryptanalysis, first is computed a GB of the ideal $I = \langle \mathcal{F} \rangle$, where $\mathcal{F}$ is the target polynomial system, using algorithms such as F_4, F_5 or ad-hoc polynomial representations and typically `DegRevLex` monomial ordering. Then, this Gröbner basis is converted into a univariate polynomial from which the set of solutions can be read off (*e.g.*, shape-position). Lastly, Cantor–Zassenhaus [29] or Shoup [30] algorithms can then be used to find the roots of this univariate polynomial, where their complexity depends on the degree of the polynomial and the size of the field.

As in the rest of this thesis, the Gröbner basis of Skyscraper's polynomial system typically yields the solution to the original system, without needing to pass through a separate univariate root-finding step.

The CICO constraint arises naturally from the Sponge mode of a compression function. Assuming a permutation-based Sponge mode, we consider the polynomial system corresponding to the permutation under a CICO constraint.

Definition: *CICO-(k_1,k_2)*

Let $F : \mathbb{K}^n \to \mathbb{K}^n$ be a function and let $k_2 < n$ be an integer and $k_1 = n - k_2$. The CICO-(k_1,k_2) problem [31] consists of finding $x \in \mathbb{K}^{k_2}$, $y \in \mathbb{K}^{k_1}$ such that $F(x||\mathbf{0}^{k_1}) = (y||\mathbf{0}^{k_2})$, where k_2 denotes the rate and k_1 the capacity of the Sponge construction.

Remark

As stated, CICO-(k_1,k_2) asks for *any* pair (x, y) satisfying the constraint, with both the input x and the output y free. Therefore, it is a search problem with generally many solutions, rather than a pre-image search against one specific/fixed digest. In fact, fixing y in advance changes the number of free variables of the polynomial system, turning the free-output search into a pre-image problem for a given digest.

The CICO systems considered in this thesis are always constructed to have finitely many solutions over $\mathbb{F}_p$.

3 Skyscraper

This chapter introduces the Arithmetization-Oriented hash function Skyscraper, which has been chosen as the cipher target for this thesis. We will first explain the inner details of its design, then we will analyze it algebraically, *i.e.*, we will instantiate the CICO problem for Skyscraper's permutation, we will derive the resulting polynomial system and we will report the authors' results of computing a Gröbner basis of it. This chapter is fundamental to precisely describe the problem which we will address in the later Chapter 4.

3.1 Skyscraper Hash Function

Skyscraper [8] is an Arithmetization-Oriented hash function designed for large prime fields $\mathbb{F}_p$ with $p \geq 2^{248}$. Its sponge construction follows the standard absorb-and-squeeze paradigm, *i.e.*, message blocks are absorbed via addition in $\mathbb{F}_p$ into the rate part of the state, and the internal permutation $\mathcal{P}$ is applied after each absorption. The hash output is extracted from the rate part during the squeezing phase.

3.1.1 State and Sponge Construction

The state consists of two branches $(x_L, x_R) \in \mathbb{F}_p^2$, where:

- x_R is the rate, thus the part into which message blocks are absorbed and from which output is squeezed;
- x_L is the capacity, thus the part that is never directly written to or read from by the outside world and whose secrecy is what the sponge's security proof relies on.

This assignment is what composes the CICO instantiation of Skyscraper, which fixes x_L and reads x_R as the value under attack. The sponge has capacity c and rate r and each call to $\mathcal{P} : \mathbb{F}_p^2 \to \mathbb{F}_p^2$ constitutes one permutation application. The classical Sponge construction [32] is depicted in Figure 3.1.

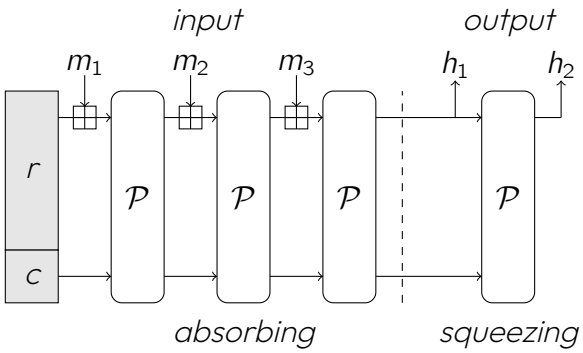


Figure 3.1: Classical absorb-and-squeeze sponge construction.

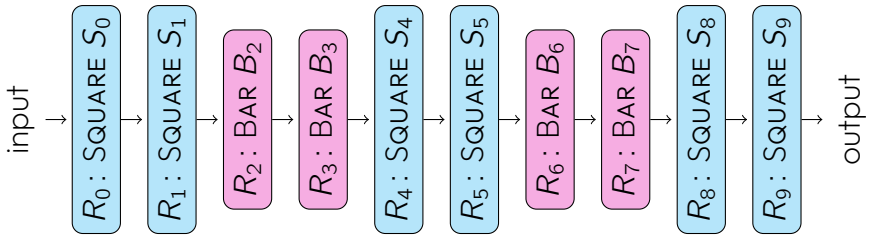


Figure 3.2: Skyscraper's permutation function $\mathcal{P}$ for $R = 10$ rounds.

3.1.2 Permutation $\mathcal{P}$

The permutation alternates between two types of rounds:

- **SQUARE rounds** that apply the function S_i;
- **BAR rounds** that apply the function B_i;

where

- R is the number of total rounds for a permutation;
- $0 \leq i < R$ is the index of the i-th round R_i;
- S_i, B_i are respectively the SQUARE and BAR functions for round R_i with i-th round constant γ_i.

Both S and B are Feistel-like functions, which ensures invertibility. The standard number of rounds is $R = 10$, where the SQUARE rounds are $R_0, R_1, R_4, R_5, R_8, R_9$, while the BAR ones are R_2, R_3, R_6, R_7, *i.e.*, the function structure is SQUARE, SQUARE, BAR, BAR, SQUARE, SQUARE, BAR, BAR, SQUARE, SQUARE, as illustrated in Figure 3.2. The reduced-round variant with $R = 6$, which is used for the experiments of Chapter 6, follows the same alternation truncated to the first six rounds, *i.e.*, the pattern SQUARE, SQUARE, BAR, BAR, SQUARE, SQUARE (R_0, R_1 SQUARE; R_2, R_3 BAR; R_4, R_5 SQUARE). In this way both variants apply the same number of SQUARE rounds before the first BAR round and the same number of BAR layers per two SQUARE rounds, differing only in how many times the middle blocks are repeated.

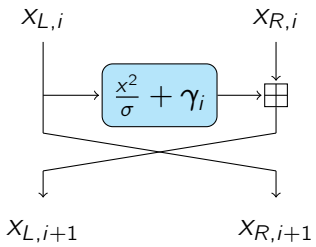


Figure 3.3: SQUARE function S.

SQUARE Function

Definition: *SQUARE*

The function $S : \mathbb{F}_p^2 \to \mathbb{F}_p^2$ is defined as

$$S_i(x_{L,i}, x_{R,i}) = \left(x_{R,i} + \frac{x_{L,i}^2}{\sigma} + \gamma_i,\ x_{L,i}\right),$$

where i is the index of round R_i, thus $x_{L,i}, x_{R,i}$ denotes respectively the left and right input for round R_i. Moreover, $\gamma_i \in \mathbb{F}_p$ is a pseudo-random round constant and σ is a fixed constant: $\sigma = 2^{256} \pmod p$ if Montgomery modular multiplication is applied, $\sigma = 1$ otherwise.

The squaring operation is very cheap inside ZK arithmetic circuits and provides good non-linearity. The function is depicted in Figure 3.3.

BAR Function and the Lookup Table

Definition: *BAR*

The function $B : \mathbb{F}_p^2 \to \mathbb{F}_p^2$ is defined as

$$B_i(x_{L,i}, x_{R,i}) = \left(x_{R,i} + B'(x_{L,i}) + \gamma_i,\ x_{L,i}\right),$$

where $B' : \mathbb{F}_p \to \mathbb{F}_p$ is the nonlinear lookup component, kept under the same notation as the SQUARE function S_i.

In Skyscraper, the S-box table used in B' is derived from the one introduced in Monolith [7] and is reused in the same small-domain lookup style over $\mathbb{F}_{2^\omega}$. For $\omega = 4$, the concrete table used in this thesis's experiments is given in Table 3.1.

Table 3.1: The $\omega = 4$ lookup table $T : \mathbb{F}_{16} \to \mathbb{F}_{16}$ used in the experiments of Chapter 6.

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$T(x)$	0	2	4	7	8	10	14	12	1	11	5	6	13	3	9	15

The function B' implements a *lookup table design* and operates as follows:

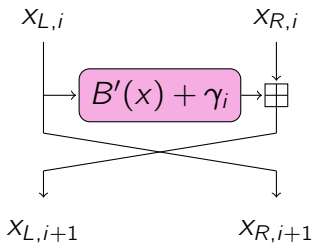


Figure 3.4: BAR function B.

1. **Decompose.** Write $X \in \mathbb{F}_p$ as a sequence of m chunks $(z_0, z_1, \ldots, z_{m-1})$, with each chunk value z_j in $\{0, \ldots, 2^\omega - 1\} \subset \mathbb{F}_p$, so that $x = \sum_{j=0}^{m-1} z_j \cdot 2^{\omega \cdot (m-1-j)}$. Each chunk is conceptually an element of $\mathbb{F}_{2^\omega}$, but it is represented as this integer subset of $\mathbb{F}_p$.

2. **Left rotation.** Apply a cyclic left rotation by $\lfloor m/2 \rfloor$ positions to the chunk sequence:

$$(z_0, z_1, \ldots, z_{m-1}) \mapsto (z_{\lfloor m/2 \rfloor}, \ldots, z_{m-1}, z_0, \ldots, z_{\lfloor m/2 \rfloor - 1}).$$

3. **Lookup-table application.** Apply the same lookup table $T : \mathbb{F}_{2^\omega} \to \mathbb{F}_{2^\omega}$ independently to each chunk, *i.e.*, $\tilde{z}_j = T(z'_j)$ for each rotated chunk z'_j.

4. **Recompose.** Assemble the output chunks back into a single $\mathbb{F}_p$ element, so $Y = \sum_{j=0}^{m-1} \tilde{z}_j \cdot 2^{\omega \cdot (m-1-j)}$.

The BAR function is shown in Figure 3.4, while B' is illustrated in Figure 3.5.

Since B', and specifically the lookup table T, is the object that Chapter 4 targets, we remark here that T is a fixed, publicly known lookup table. The security of T, and thus the difficulty of attacking it, comes from having no compact algebraic description over $\mathbb{F}_p$ other than an interpolation polynomial of large degree, and not from T being secret.

Security Claims

In [8, Section 2.4], Skyscraper's authors claim that, for all primes larger than 248 bits, at least 124 bits of security are guaranteed against both collision and pre-image attacks on the sponge hash function. In particular, they prove that any linear approximation of the BAR function holds with probability too low to be exploitable in a classical linear cryptanalysis attack, *i.e.*, a distinguishing attack against $\mathcal{P}$ itself viewed as a black-box function of its input. This claim is not related with the attack developed in this thesis because our proposed affine replacement is not used to build a statistical distinguisher against $\mathcal{P}$, but as an alternative algebraic representation of the lookup table T inside an already-instantiated CICO polynomial system. Therefore, this affine replacement is used to make verification cheaper for each case of the exact case-split.

The algebraic properties of the construction were also analyzed directly by Skyscraper's designers by running Gröbner basis computation on the CICO instantiation. For transparency, we have reported in Table 3.2 the running times from [8, Table 16] for $\omega = 4$ and $R = 6$ rounds. These results are used again in Chapter 6 as a baseline comparison for our hybrid attack. As discussed in Section 6.1, they were obtained on different hardware than the results of this thesis, so the comparison should be read as indicative rather than as a controlled benchmark.

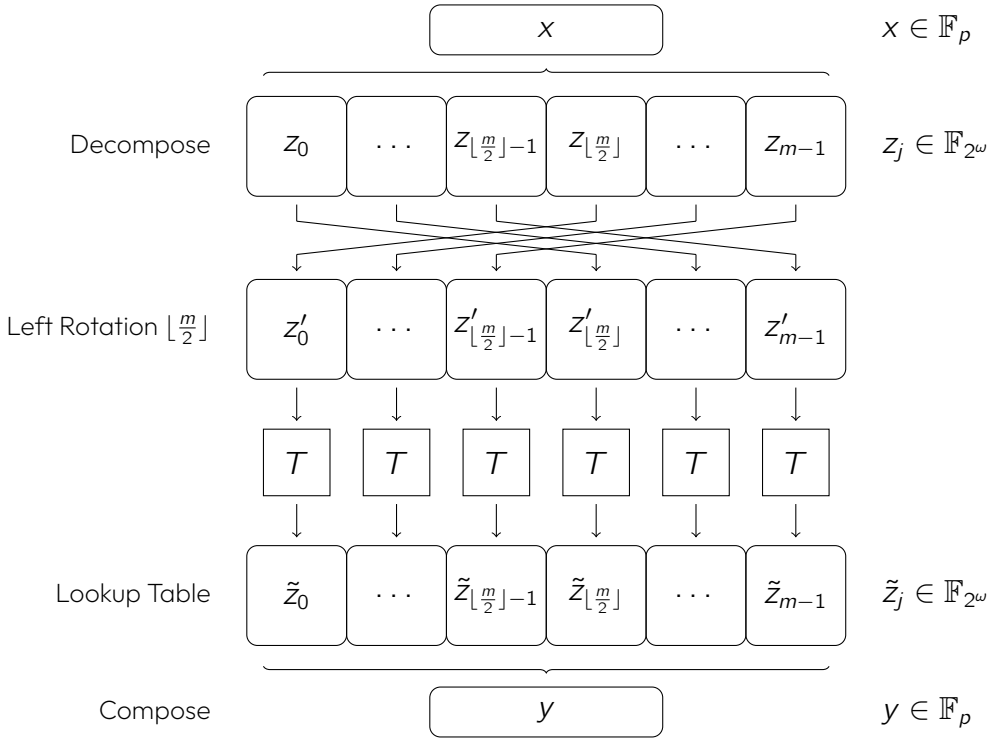


Figure 3.5: B' function over m chunks, each of ω bits.

Table 3.2: Gröbner basis cryptanalysis results against CICO Skyscraper for $\omega = 4$, $R = 6$, from [8, Table 16].

p	m	time (s)
37	2	1
109	2	2
163	2	3
191	2	4
587	3	243
1237	3	7 550
1361	3	18 974
1399	3	18 912
2297	3	51 103
2953	3	123 706

In Table 3.2, the running time grows sharply with m[1] even though ω is held fixed. Within a fixed m, the running time still grows substantially with p, even though ω, and hence the degree of the interpolation polynomial of T, does not depend on p at all. The second observation already hints that degree alone does not explain the cost, but rather how the chunk-level equations interact as p grows. The original paper has not reported results for $m > 3$ nor for $\omega = 8$, and one of the results of this thesis (Section 6.2) is that we have been able to solve instances that were previously untested, precisely by attacking the bottleneck identified in this chapter.

[1] the number of chunks in the Bar-function

3.2 Skyscraper-CICO

We model the cryptanalytic attack as an instance of the generic CICO problem, specialized to Skyscraper's two-branch state, with capacity x_L and rate x_R.

Definition: *Skyscraper-CICO problem*

Given the Skyscraper permutation $\mathcal{P}\colon \mathbb{F}_p^2 \to \mathbb{F}_p^2$, the *CICO problem* consists of finding all pairs $(x_R, y_L) \in \mathbb{F}_p^2$ such that

$$\mathcal{P}(0, x_R) = (y_L, 0),$$

i.e., the left branch (capacity) of the input and the right branch (rate) of the output are fixed to 0.

In the definition above, both x_R and y_L are free variables, but y_L is determined by x_R, so the solutions are in one-to-one correspondence with the values x_R for which the right output of $\mathcal{P}(0, x_R)$ is 0. Since $\mathcal{P}$ is a permutation, it cannot be used to state a collision on $\mathcal{P}$ itself, as $\mathcal{P}(0, x_R) = \mathcal{P}(0, x'_R)$ implies $x_R = x'_R$. Both attacks below are therefore stated on the right output of the permutation, $h(x_R) = y_R$ where $(y_L, y_R) = \mathcal{P}(0, x_R)$, which is the value a sponge squeezes from the rate:

- **Collision search.** Two distinct values $x_R \neq x'_R$ with $h(x_R) = h(x'_R)$, where the left outputs y_L, y'_L are unconstrained and in general different. Skyscraper-CICO, which asks for $h(x_R) = 0$, is the special case in which the common value is 0: any two of its solutions form a collision on the sponge.
- **Pre-image search.** Given a target digest $h \in \mathbb{F}_p$, finding $x_R \in \mathbb{F}_p$ such that $\mathcal{P}(0, x_R) = (y_L, h)$ for some y_L. This generalizes Skyscraper-CICO from the fixed target 0 to an arbitrary target and is the problem solved by the experiments of Chapter 6. The polynomial system keeps the same equations, with y_L left as an unconstrained output, and the equation defining y_R becomes $x_{L,9} = h$.

A key observation, reported by Skyscraper's designers [8, Section 4.3.2] and confirmed by our own experiments, is that the Gröbner basis computation of the ideal encoding the Skyscraper-CICO system for a *pre-image* search returns the solution of the instance directly, without any univariate root-finding or back-substitution step. This behavior is governed by the degenerate case for degree-1 of the system in shape position. By contrast, in the case of *collision search*, the Skyscraper-CICO system assumes a general shape position.

3.3 Polynomial Model

Each round R_i of the permutation $\mathcal{P}$ contributes polynomial equations in the state variables. Introducing intermediate variables for each round's input and output, the full permutation is encoded as a polynomial system $\mathcal{F} = \{f_1, \ldots, f_k\}$ over $\mathbb{F}_p$:

- the SQUARE round equations are low-degree, in particular, are kept quadratic via the introduction of intermediate auxiliary variables;
- the BAR round equations are where the degree and density of the system explode.

Let $(x_{L,i}, x_{R,i})$ respectively be the left and right inputs to the Skyscraper permutation for round i, where $0 \leq i < R$, and let (y_L, y_R) respectively be the final left and right outputs of the permutation. For the function B', we denote with X and Y its input and output, and with z_j, $\tilde{z}_j$ respectively the post-decompose and pre-compose chunks, where $0 \leq j < m$, and the chunk at index 0 contains the most significant bits.

SQUARE function

The SQUARE function is encoded as

$$\mathcal{F}_S(x_{L,i}, x_{R,i}) = \begin{cases} x_{L,i+1} = x_{R,i} + \frac{x_{L,i}^2}{\sigma} + \gamma_i \\ x_{R,i+1} = x_{L,i} \end{cases}.$$

B′ function

To define the polynomial system for the BAR function, we first need to define it for B':

- *decomposition* constraints: $X_i = \sum_{j=0}^{m-1} z_j \cdot 2^{\omega \cdot (m-1-j)}$;
- *range* constraints: $\prod_{w=0}^{2^\omega - 1}(z_j - w)$ to make each chunk variable z_j fit into 2^ω bits, *i.e.*, take a value in $\{0, \ldots, 2^\omega - 1\}$ (this is the field-membership polynomial);
- *rotation* and *lookup-table* T constraints: $\tilde{z}_j = T(z_{(j+\lfloor m/2 \rfloor) \bmod m})$;
- *composition* constraints: $Y_i = \sum_{j=0}^{m-1} \tilde{z}_j \cdot 2^{\omega \cdot (m-1-j)}$.

$$\mathcal{F}_{B'}(x_{L,i}) = \begin{cases} X_i = \sum_{j=0}^{m-1} z_j \cdot 2^{\omega \cdot (m-1-j)} \\ \prod_{w=0}^{2^\omega - 1}(z_j - w) \\ \tilde{z}_j = T(z_{(j+\lfloor m/2 \rfloor) \bmod m}) \\ Y_i = \sum_{j=0}^{m-1} \tilde{z}_j \cdot 2^{\omega \cdot (m-1-j)} \end{cases}.$$

BAR function

The BAR function is then

$$\mathcal{F}_B(x_{L,i}, x_{R,i}) = \mathcal{F}_{B'}(x_{L,i}) \cup \begin{cases} x_{L,i+1} = Y_i + \gamma_i + x_{R,i} \\ x_{R,i+1} = x_{L,i} \end{cases}.$$

Final Polynomial System

Definition: *Polynomial model for Skyscraper*

$$\mathcal{F} := \bigcup_{i=0}^{1} \mathcal{F}_S(x_{L,i}, x_{R,i}) \cup \bigcup_{i=2}^{3} \mathcal{F}_B(x_{L,i}, x_{R,i}) \cup \bigcup_{i=4}^{5} \mathcal{F}_S(x_{L,i}, x_{R,i}) \cup \bigcup_{i=6}^{7} \mathcal{F}_B(x_{L,i}, x_{R,i}) \bigcup_{i=8}^{9} \mathcal{F}_S(x_{L,i}, x_{R,i}).$$

The final polynomial system encoding the entire Skyscraper permutation is the concatenation of two SQUARE and two BAR polynomial systems, repeated until the number of rounds equals $R = 10$:

$$
\begin{cases}
x_{L,1} = x_{R,0} + \frac{x_{L,0}^2}{\sigma} + \gamma_0 \\
x_{R,1} = x_{L,0} \\
\\
x_{L,2} = x_{R,1} + \frac{x_{L,1}^2}{\sigma} + \gamma_1 \\
x_{R,2} = x_{L,1} \\
\\
x_{L,2} = \sum_{j=0}^{m-1} z_j \cdot 2^{\omega \cdot (m-1-j)} \\
\prod_{w=0}^{2^\omega - 1} (z_0 - w) \\
\dots \\
\prod_{w=0}^{2^\omega - 1} (z_{m-1} - w) \\
\tilde{z}_0 = T(z_{(0+\lfloor m/2 \rfloor) \bmod m}) \\
\dots \\
\tilde{z}_{m-1} = T(z_{(m-1+\lfloor m/2 \rfloor) \bmod m}) \\
Y_2 = \sum_{j=0}^{m-1} \tilde{z}_j \cdot 2^{\omega \cdot (m-1-j)} \\
x_{L,3} = Y_2 + \gamma_2 + x_{R,2} \\
x_{R,3} = x_{L,2} \\
\\
\dots \\
\\
x_{L,9} = x_{R,8} + \frac{x_{L,8}^2}{\sigma} + \gamma_8 \\
x_{R,9} = x_{L,8} \\
y_L = x_{R,9} + \frac{x_{L,9}^2}{\sigma} + \gamma_9 \\
y_R = x_{L,9}
\end{cases}
$$

In particular, for the pre-image search we have $x_{L,0} = 0$ and $x_{R,0}$ is the hash input, while y_L is an unconstrained output and $y_R = h$ is the digest.

3.3.1 Gröbner Basis Complexity

Having derived the full system $\mathcal{F}$, we now examine why the BAR rounds dominate the cost of computing a Gröbner basis of $\langle \mathcal{F} \rangle$.

The main source of complexity arises from the BAR rounds. Each call to B' introduces the lookup table T on each ω-bit chunk. Encoding T as a polynomial over $\mathbb{F}_p$ via Lagrange interpolation on the set $\{0, 1, \dots, 2^\omega - 1\} \subset \mathbb{F}_p$ gives a polynomial of degree $2^\omega - 1$. For $\omega = 4$, such a polynomial has degree 15.

It is easy to mistake such an interpolation polynomial for something more structured than it is, so we report an example of its actual coefficients in Table 3.3, for the case of prime field $\mathbb{F}_{587}$ of Table 3.1, with $\omega = 4$.

Table 3.3: Coefficients of the degree-15 Lagrange interpolation polynomial $\sum_{k=0}^{15} \alpha_k z^k$ of the $\omega = 4$ lookup table of Table 3.1 over $\mathbb{F}_{587}$.

α_0	α_1	α_2	α_3	α_4	α_5	α_6	α_7	α_8	α_9	α_{10}	α_{11}	α_{12}	α_{13}	α_{14}	α_{15}
0	231	456	550	263	92	184	447	508	547	391	285	258	306	169	11

The interpolation polynomial is dense, since fifteen of its sixteen coefficients are nonzero (only the constant term vanishes, as a direct consequence of $T(0) = 0$), and these coefficients depend on both the table T and the prime p.

Moreover, the field-switch, from $\mathbb{F}_p$ to $\mathbb{F}_{2^\omega}$, further complicates the situation, as it requires including the *field-membership constraints* of Section 3.3 to assert that each chunk variable z_j is a valid ω-bit integer and thus belongs to $\{0, 1, \ldots, 2^\omega - 1\}$. For $\omega = 4$, this means introducing m polynomials of degree $2^\omega = 16$, one per chunk.

For $m = 3$ chunks (so X decomposes into z_0, z_1, z_2), the system of polynomials generated during one application of B' is:

$$\begin{cases} X - (16^2 \cdot z_0 + 16^1 \cdot z_1 + 16^0 \cdot z_2), & X \text{ decomposition} \\ \\ \prod_{w=0}^{15}(z_0 - w), & \mathcal{F}_{16} \text{ constraint for } z_0 \\ \prod_{w=0}^{15}(z_1 - w), & \mathcal{F}_{16} \text{ constraint for } z_1 \\ \prod_{w=0}^{15}(z_2 - w), & \mathcal{F}_{16} \text{ constraint for } z_2 \\ & \text{Left-Rot}([z_0,z_1,z_2]) \to [z_1,z_2,z_0] \\ \sum_{w=0}^{15}(\alpha_w z_1^w) - \tilde{z}_0, & \tilde{z}_0 = T(z_1) \\ \sum_{w=0}^{15}(\alpha_w z_2^w) - \tilde{z}_1, & \tilde{z}_1 = T(z_2) \\ \sum_{w=0}^{15}(\alpha_w z_0^w) - \tilde{z}_2, & \tilde{z}_2 = T(z_0) \\ \\ (16^2 \cdot \tilde{z}_0 + 16^1 \cdot \tilde{z}_1 + 16^0 \cdot \tilde{z}_2) - Y & Y \text{ composition} \end{cases}$$

where $\alpha_w \in \mathbb{F}_p$, $0 \leq w \leq 15$, are the coefficients of T's interpolation polynomial, like those just presented in Table 3.3.

Moreover, we report actual measurements of the Gröbner basis computation's behavior on a small instance of the B' system above for $\omega = 2$, $m = 2$ chunks and $p = 13$:

Table 3.4: Reduced Gröbner basis of the isolated B' polynomial system for $\omega = 2$, $m = 2$, $p = 13$, under two monomial orderings, respectively `Lex` and `DegLex`.

Ordering	# generators	Degrees of generators	Max. degree
`Lex`	5	12, 12, 12, 12, 13	13
`DegLex`	11	5, 4, 4, 4, 4, 3, 3, 3, 3, 3, 1	5

4 Affine Cover Attack

Chapter 3 presented the specification of the Skyscraper hash function, including an evaluation of the cost of the Gröbner basis computation for Skyscraper-CICO. This cost is dominated by the Bar rounds and depends on two causes, namely the interpolation polynomial of the lookup table T, which has degree $2^\omega - 1$, and the need to introduce polynomial-membership constraints for each chunk variable. This chapter presents this thesis's core contribution, which consists of replacing T by an affine function of degree 1. This replacement affect not only the degree of T, but also restricts the possible range values of each chunk to a smaller set. The affine function is chosen such that is an exact case-split of T and the total of affine functions needed to cover T is low. However, the main drawback is that with this solution, to perform a pre-image search, we need to enumerate over the possible cases of affine functions.

4.1 Piecewise Affine Approximation

The key observation is that T admits a geometric interpretation, *i.e.*, the table $T : \{0, \ldots, 2^\omega - 1\} \to \{0, \ldots, 2^\omega - 1\}$ can be viewed as a finite set of points $(x, T(x))$ in $\mathbb{F}_p^2$. Although T is globally a polynomial of degree $2^\omega - 1$, *locally*, many of these points lie on lines, as can be seen from the scatter plot of the lookup table for $\omega = 4$ and $\omega = 8$ in Figure 4.1. That is, there exist affine functions $L_k(x) = \alpha_k x + \beta_k$ (with $\alpha_k, \beta_k \in \mathbb{F}_p$) such that each L_k agrees with T on some subset $\mathcal{S}_k \subseteq \{0, \ldots, 2^\omega - 1\}$.

This is possible, and should not be overstated as a special property of T, since any two points over $\mathbb{F}_p$ determine a line, so any two-element subset of the graph of T trivially admits an affine function agreeing with it exactly. Therefore, there always exists a cover of the entire domain of T by affine pieces. What is not automatic, and is a genuine design choice discussed below, is finding a cover with *few* pieces and with *large, well-separated* subsets $\mathcal{S}_k$, since both properties make the resulting case-split attack practical rather than merely possible.

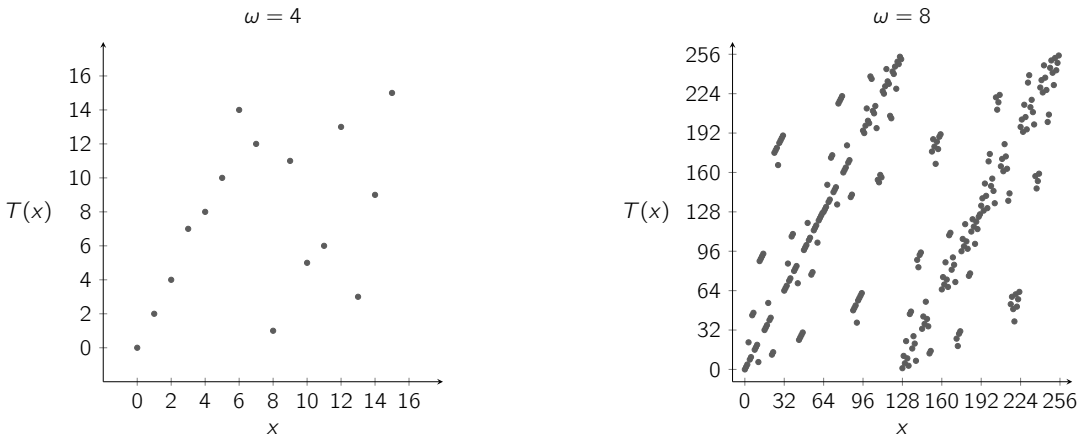


Figure 4.1: Scatter plot of $(x, T(x))$ for different ω.

Definition: *Affine agreement piece*

An *affine agreement piece* for the lookup table T is a pair $(\alpha, \beta) \in \mathbb{F}_p^2$ together with a subset $\mathcal{S} \subseteq \{0, \ldots, 2^\omega - 1\}$ such that

$$T(x) = \alpha x + \beta \quad \forall x \in \mathcal{S}.$$

A finite collection $\{(L_1, \mathcal{S}_1), \ldots, (L_\ell, \mathcal{S}_\ell)\}$ of affine agreement pieces is a *cover* of T if $\mathcal{S}_1 \cup \cdots \cup \mathcal{S}_\ell = \{0, \ldots, 2^\omega - 1\}$.

What is uncertain about this affine cover is not how well each piece L_k tracks T, but simply *which* is the correct selection of pieces for a given input. This is why our hybrid attack (presented later in Section 4.5) is essentially a *guess-and-determine* attack, *i.e.*, it guesses, for each chunk, which piece of the cover the true chunk value belongs to and determines the rest by solving the obtained reduced system under that guess. The probabilistic component of this attack will be analyzed separately in Chapter 5, to answer the purely combinatorial question of which guesses to try first, given that they cannot all be tried at once. Therefore, whenever we talk about probability in the following, we refer to this search order and not to the algebraic relationship between L_k and T on $\mathcal{S}_k$.

For $\omega = 4$, analyzing the scatter plot of $(x, T(x))$ with the overlaid affine pieces shown in Figure 4.2 yields a cover by five pieces:

$$\begin{aligned} L_1 &: y = 2x, & \mathcal{S}_1 &= \{0, 1, 2, 4, 5\}, \\ L_2 &: y = \tfrac{2}{3}x + 5, & \mathcal{S}_2 &= \{3, 9, 12, 15\}, \\ L_3 &: y = x - 5, & \mathcal{S}_3 &= \{10, 11, 14\}, \\ L_4 &: y = -2x + 26, & \mathcal{S}_4 &= \{6, 7\}, \\ L_5 &: y = \tfrac{2}{5}x - \tfrac{11}{5}, & \mathcal{S}_5 &= \{8, 13\}. \end{aligned}$$

Note

From now on, with ℓ we refer to the number of pieces in the cover, which in this running example are $\ell = 5$.

This particular cover in the example, is one design choice among many possible covers for T and is not a canonical or unique. Moreover, the sets $\mathcal{S}_1, \ldots, \mathcal{S}_5$ of this cover choice

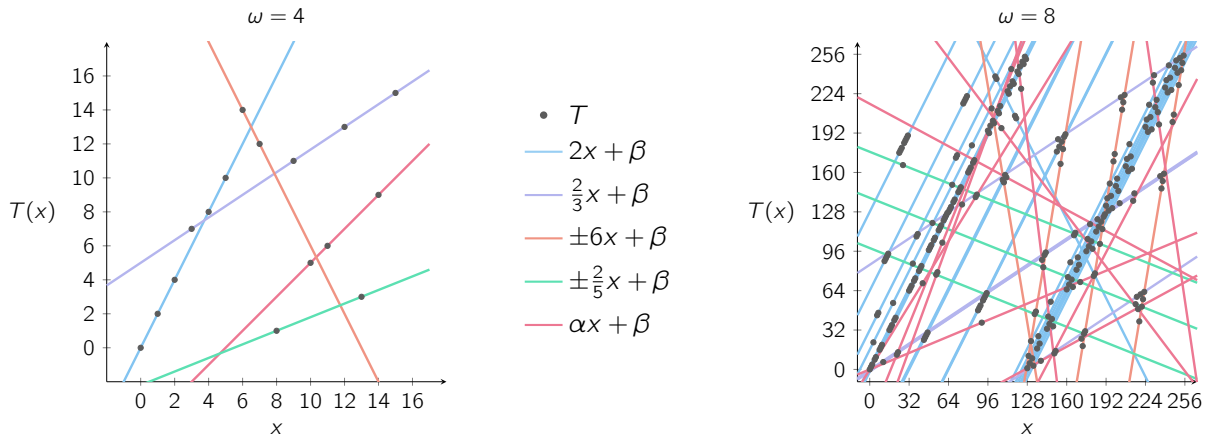


Figure 4.2: Affine cover of the scatter plot of $(x, T(x))$ (Figure 4.1) for different ω.

happen to be disjoint and to partition $\{0, \ldots, 15\}$. In the following chapter, we will describe the heuristics behind our choice of covers for $\omega = 4$ and $\omega = 8$ cases and we will also explain what are the requirements for a good cover.

4.2 Cover Selection Heuristics

Since any two points determine a line, the existence of *some* affine cover of T is trivial and gives no guidance on its own. Our criterion for picking out one cover among the many possible ones goes accordingly to the following two objectives, applied in this order:

1. **Minimize the number of pieces ℓ.** Fewer pieces mean fewer cases that the hybrid attack needs to enumerate over per chunk. This directly shrinks the size of the search space the attack must explore.

2. **Minimize the overlap between pieces.** Among the covers achieving the minimum ℓ, we prefer those whose subsets $\mathcal{S}_k$ are pairwise disjoint, or as close to disjoint as possible. The reason is that an input x lying in the overlap of two pieces $\mathcal{S}_k \cap \mathcal{S}_{k'}$ is correctly covered by *either* choice, so it does not need to be tried under both. This means that a cover with large overlaps effectively re-examines the same inputs under multiple cases, wasting search effort without covering any domain point that a lower-overlap cover of the same size would have missed.

For $\omega = 4$, the five-piece cover of our previous example was obtained by directly inspecting the scatter plot of $(x, T(x))$ against these two criteria. In particular, for the definition of T and $\omega = 4$, no smaller cover ($\ell < 5$) was found and the respective subsets of our five-piece cover are disjoint.

Finding a minimum-size cover by affine pieces is a set-cover-type problem: the candidate pieces are the maximal pieces determined by pairs of points of the graph of T and a cover selects the fewest of them whose subsets cover $\{0, \ldots, 2^\omega - 1\}$. Covering a set of points of the plane with the minimum number of lines is NP-hard in general [33]. This does not directly settle the complexity of the instances considered here, since T is the graph of a function over a small domain and the number of maximal pieces is finite. We therefore claim

no polynomial-time exact algorithm and the practical approach adopted is heuristic rather than provably optimal.

We treat $\omega = 4$ throughout most of this thesis as a didactic and testing reduction, because it is small enough to inspect and verify and it allowed us to run the full experiments reported in Chapter 6. An affine cover for the $\omega = 8$ case is shown in Figure 4.2 and listed in Appendix B, but we do not provide any guarantee that it is optimal, since a systematic search algorithm is needed for this task.

Concretely, a cover satisfying these criteria is found by brute-force search over combinations of affine pieces. Since any two points of the graph of T determine a candidate piece $(L, \mathcal{S})$, one can start from such a pair and greedily include every further point of the graph on which L also agrees with T. Therefore, the set of maximal pieces[1] is finite and can be enumerated exhaustively for small ω. However, this approach does not scale efficiently to $\omega = 8$, where the number of candidate maximal pieces grows substantially, and this motivates the need of a more systematic algorithm.

Two directions could improve on brute-force search by using information that the current cover-selection step ignores.

- **An explicit covering algorithm for general ω.** Selecting a minimum-size, minimum-overlap affine cover of a finite function's graph is an instance of a general combinatorial covering problem. Thus, a dedicated algorithm exploiting this structure, rather than brute-force enumeration of maximal pieces, is needed to scale to $\omega = 8$ and beyond, and it is left as future work as specified in Section 7.3.

- **Ordering pieces by size and by cross-layer linkage.** Once a cover is fixed, two more heuristics can be taken into consideration to improve the *order* in which our hybrid attack tries combinations of pieces. The first is trivially trying large-$|\mathcal{S}_k|$ pieces first. The second is to take into consideration the cross-layer linkage between SQUARE and BAR, rather than treating the choice at each layer as independent of the others. This implies preferentially combining a choice of piece at one BAR layer with choices at the next layer whose subsets $\mathcal{S}_{k'}$ are likely to contain the actual chunk values that L_k would *output* for inputs in $\mathcal{S}_k$. Developing and evaluating this second heuristic concretely has been also flagged as future work in Section 7.3.

4.3 Cost Model

Before deriving the reduced polynomial system, we state the cost model that justifies this approach, since it is what Section 4.5 and Chapter 5 both try to optimize, in different ways. The total cost of the attack is given by

$$\text{Total attack cost} \;=\; \big(\text{number of cases tried}\big) \times \big(\text{cost of Gröbner basis computation}\big).$$

The baseline attack used by the authors of Skyscraper is the base case of this cost. It has exactly one case, which represents all possible values simultaneously, and the cost of the Gröbner basis computation is maximal, since that case must use the full degree-$(2^\omega - 1)$ interpolation polynomial of T. The solution presented in this thesis by means of an affine cover of T, balances the total cost equation differently. Namely, it multiplies the number of cases

[1] pieces that cannot be enlarged without ceasing to agree with T

that might need to be tried (up to ℓ^m for m chunk positions) but collapses the per-case Gröbner basis computational cost, since each case's reduced system has only degree-1 equations and range constraints of degree at most $\max_k |\mathcal{S}_k| \leq 2^\omega - 1$. The method pays off when the reduction in the Gröbner basis computational cost outweighs the number of cases actually needed in practice. This is why the cover-selection criteria (few pieces of degree 1) and the enumeration order, which keeps the number of cases tried before success far below the worst case ℓ^q, act directly on the two factors of this equation.

4.4 Reduced Polynomial System

For notational convenience, we denote by $\mathcal{S}_{k,j}$ the subset associated with the chunk z_j under case k, and by $L_{k,j}$ the corresponding affine function. Under the assumption that the chunk value z_j lies in $\mathcal{S}_{k,j}$, the function $T(z_j)$ is replaced by $\alpha_{k,j} z_j + \beta_{k,j}$, with no loss of exactness on $\mathcal{S}_{k,j}$. Denoting with $\rho = \lfloor m/2 \rfloor$ the fixed rotation offset, the resulting polynomial system encoding the B' function under case k becomes:

$$\begin{cases} X = \sum_{j=0}^{m-1} z_j \cdot 2^{\omega\cdot(m-1-j)} & (0) \\ \prod_{s\in\mathcal{S}_{k,0}} (z_0 - s) = 0 & (1) \\ \dots & \\ \prod_{s\in\mathcal{S}_{k,j}} (z_j - s) = 0 & (j{+}1) \\ \dots & \\ \prod_{s\in\mathcal{S}_{k,m-1}} (z_{m-1} - s) = 0 & (m) \\ Y = \sum_{i=0}^{m-1} \left(\alpha_{k,\,(i+\rho) \bmod m}\, z_{(i+\rho) \bmod m} + \beta_{k,\,(i+\rho) \bmod m}\right) \cdot 2^{\omega\cdot(m-1-i)} & (m{+}1) \end{cases}$$

where

- equation (0) decomposes the input $X \in \mathbb{F}_p$ into m chunks $z_j \in \mathbb{F}_{2^\omega}$;
- equations (1) through (m) enforce that each z_j belongs to its assumed subset $\mathcal{S}_{k,j}$, *i.e.*, the range constraints have degree $|\mathcal{S}_{k,j}|$ rather than the full 2^ω, since we only need to assert membership in the smaller assumed subset;
- equation $(m{+}1)$ expresses the recomposition into the output $Y \in \mathbb{F}_p$, where position i of the output receives the rotated input chunk at position $(i{+}\rho) \bmod m$, with T replaced by the affine $L_{k,\,(i+\rho) \bmod m}$ and with the positional weight $2^{\omega\cdot(m-1-i)}$ attached to the *output* position i.

Comparing this reduced system against the full polynomial model of Section 3.3, highlights that the recomposition equation is now linear, where previously was a sum of degree-$(2^\omega - 1)$ terms coming from T's interpolation polynomial Moreover, the range constraints equations from (1) to (m), were all previously fixed at degree 2^ω, while now have degree $|\mathcal{S}_{k,j}| \leq 2^\omega - 1$ (at most 5 for the cover of $\omega = 4$), which is dictated entirely by which piece is assumed. Consequently, none of the high-degree dense generators measured in Table 3.4 is present because the reduced system's generators have degree at most $\max_j |\mathcal{S}_{k,j}| \leq 2^\omega - 1$.

By the Macaulay bound, this lowers the ceiling on the degrees a Gröbner basis computation over the reduced system can reach. The two systems do not have the same number of variables and equations (the reduced one has no variables $\tilde{z}_j$ and no interpolation equations), so the bound cannot be compared through a single formula in n. What drives it is the sum of the degrees of the generators of the B' subsystem. In the full model this sum is $m\,2^\omega$ for the

range constraints, $m(2^\omega - 1)$ for the interpolation equations and 2 for the decomposition and composition equations, *i.e.*, $m(2^{\omega+1} - 1) + 2$, which equals $31m + 2$ for $\omega = 4$. In the reduced model the interpolation equations disappear, since the recomposition equation is linear, and the range constraints have degree $|\mathcal{S}_{k,j}|$, so the sum is $\sum_j |\mathcal{S}_{k,j}| + 2 \leq 5m + 2$ for the cover of $\omega = 4$. The Macaulay bound is not tight and should not be read as a precise prediction of the speedup, but it is a first indication that the affine cover is more efficient than the interpolation polynomial. The actual experimental measurement speedups will be reported in Chapter 6 (Section 6.2) and range from roughly $2\times$ to close to three orders of magnitude depending on the instance.

4.5 Hybrid Attack

Until now, we assumed that the chunk value z_j lies in a specific subset $\mathcal{S}_{k,j}$, but this is not what we know in advance, since z_j is itself part of what we are solving for. Since we cannot deterministically choose the correct piece $L_{k,j}$ for each chunk of each BAR-round application, the reduced system is turned into a full attack:

1. for each candidate case k in some enumeration order, substitute the corresponding pieces into $\mathcal{F}$ to obtain the reduced system $\mathcal{F}'_k$;
2. compute a Gröbner basis of $\mathcal{F}'_k$: if it is the trivial ideal $\langle 1 \rangle$, discard k and move to the next case;
3. otherwise, extract from the Gröbner basis the candidate values of $x_{R,0}$ in $\mathbb{F}_p$ and verify each of them by evaluating the permutation, *i.e.*, by checking that $\mathcal{P}(0, x_{R,0})$ has the required right output. Then return the first verified value.

The actual implementation of this loop, together with the enumeration routine and the affine cover data it draws on, is reported verbatim in Appendix A.

Correctness. Let $x^\star$ be a solution of the original problem and let $k^\star$ be the case whose pieces contain, at every BAR layer and for every chunk, the true chunk value of $x^\star$. Such a case exists because the subsets $\mathcal{S}_j$ of a cover jointly contain all the chunk values. By exactness of the pieces on $\mathcal{S}_{k,j}$, the point of the model corresponding to $x^\star$ satisfies every equation of $\mathcal{F}'_{k^\star}$, hence $V(\langle \mathcal{F}'_{k^\star} \rangle) \neq \varnothing$ and therefore $1 \notin \langle \mathcal{F}'_{k^\star} \rangle$. The case containing a solution is thus never discarded at step 2, so the case-split itself loses no solution, provided that the enumeration contains $k^\star$. The only restriction applied by the implementation, the head restriction of Section 4.6, never removes $k^\star$, since the true most significant chunk of any $x < p$ is at most d_0, the leading digit of p, and therefore lies in a piece that contains a value $\leq d_0$.

Conversely, a non-trivial Gröbner basis does not certify a genuine solution, for two reasons. First, by the Nullstellensatz it only guarantees a point over $\overline{\mathbb{F}_p}$, which need not lie in $\mathbb{F}_p$. Second, $\mathcal{F}'_k$ is a relaxation of the permutation. For instance, the decomposition equation is an identity in $\mathbb{F}_p$, so chunk tuples whose integer value is at least p are also admissible, and $\mathcal{F}'_k$ may therefore have points even for a case k that does not match the true chunk values. Soundness, *i.e.*, the absence of false positives, is therefore obtained only through the explicit verification of step 3, which evaluates $\mathcal{P}$ on the candidate. The extraction routine in Appendix A handles the two shapes of reduced Gröbner basis that occurred in the experiments (a linear polynomial in $x_{R,0}$, or a univariate polynomial in another variable followed by a linear

one in $x_{R,0}$) and treats any other shape as "no solution". Completeness of the implementation is therefore conditional on extracting every $\mathbb{F}_p$-root from non-trivial bases, which is not proved here.

The search space over cases is exponential in $q = m \cdot R_{\text{BAR}}$, *i.e.*, the total number of chunks across all BAR-round applications and in the standard Skyscraper parametrization for $R = 10$ rounds we have $R_{\text{BAR}} = 4$, thus $q = m \cdot 4$. On the other hand, since each individual Gröbner basis computation operates on the reduced system, we have that each iteration is vastly cheaper than the single Gröbner basis computation of the baseline attack. This makes the enumeration order matter, because an order that visits high-agreement cases first should on average require far fewer iterations before success than a naive one (*e.g.*, lexicographic in k) that has no reason to prefer one case over another. The probabilistic model needed to rank cases by estimated agreement rate is developed in Chapter 5. However, in the implementation evaluated in Chapter 6, this ranking is not applied in full, because we have only implemented the gatekeeper restriction of Section 4.6 on the most-significant chunk of each BAR layer. This means that, in our coded attack, chunk positions and the combination of choices across the R_B BAR layers are still enumerated in plain lexicographic order over piece indices, rather than sorted by the computed agreement rate $\mathbb{P}_{\text{trail}}$ of Chapter 5. Fully sorting the enumeration by $\mathbb{P}_{\text{trail}}$, rather than only gating on the most-significant chunk, is left as future work.

4.6 B' Structure Exploitation

Beyond ranking the enumeration by estimated likelihood, a second, complementary improvement consists in pruning some cases by exploiting a structural property of B' itself, independent of any probabilistic estimate. It is the case that not every combination of cover pieces is *reachable* as a valid output of B', because of how B' moves values between $\mathbb{F}_p$ and $\mathbb{F}_{2^\omega}$.

We recall that B' function job is to decompose the input $X \in \mathbb{F}_p$ into m chunks in $\mathbb{F}_{2^\omega}$, then applies T and a rotation to them, and finally recomposes the result into $\mathbb{F}_p$. The difficulty is that going from $\mathbb{F}_p$ to $\mathbb{F}_{2^\omega}$, and vice versa, means moving between spaces of different sizes, since an input in $\mathbb{F}_p$ is decomposed into the strictly larger space $\mathbb{F}_{2^\omega}^m$ of size $(2^\omega)^m \geq p$. This is caused by the definition of the number of chunks m itself, since is computed so that all the chunks together, each of ω bits, suffice to represent any element of $\mathbb{F}_p$. Because of this difference in size, the decompose and compose functions are surjective, in particular:

- the decomposition of X cannot achieve some values that a naive enumeration over $\mathbb{F}_{2^\omega}^m$ would;
- the composition of the output Y induces a *compression* effect on the values, where multiple chunk-tuples in $\mathbb{F}_{2^\omega}^m$ may recompose to the same element of $\mathbb{F}_p$.

This behavior is illustrated in Figure 4.3, which have been measured on an exhaustive evaluation on all primes with $m = 4$ for $\omega = 4$. The distribution of points in the figure shows that the closer p is to $(2^\omega)^m$, the lower the compression is and the fewer chunk-tuples are lost. Conversely, when p is much smaller than $(2^\omega)^m$ (equivalently, when p is closer to $(2^\omega)^{m-1}$), a large fraction of the $(2^\omega)^m$ chunk-tuples do not correspond to any valid input at all.

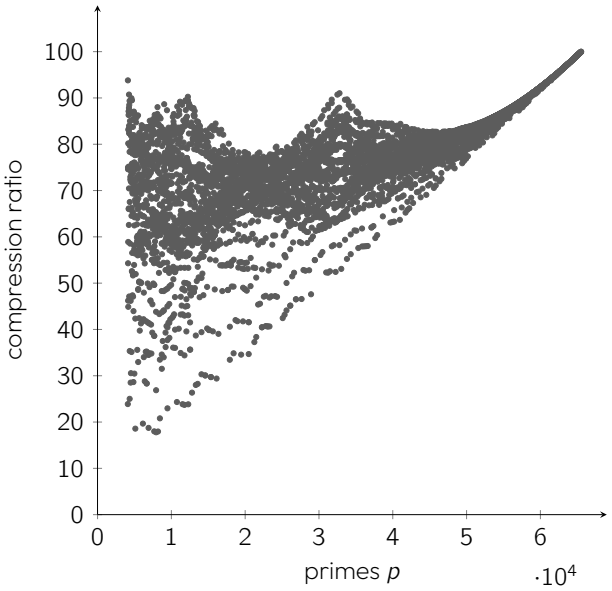


Figure 4.3: Scatter plot of compression of B' function for $m = 4$ and $\omega = 4$. The y axis is the ratio between the number of outputs and the number of inputs $\left(\frac{|B'(P)|}{|P|} \cdot 100\right)$ for the corresponding p on the x axis, with $P = \{0, \ldots, p-1\}$.

Head restriction (implemented). The only part of this reasoning exploited by the evaluated implementation concerns the most significant chunk of each BAR layer. Let $d_0 = \lfloor p/2^{\omega(m-1)} \rfloor$ be the leading base-2^ω digit of p. Since the input X of a BAR function is the canonical representative of an element of $\mathbb{F}_p$, *i.e.*, $0 \leq X < p$, its most significant chunk satisfies $z_0 \leq d_0$. A piece whose subset $\mathcal{S}_k$ contains no value $\leq d_0$ can therefore never cover the true value of z_0 and is excluded from the enumeration for that chunk. Thus, the pieces kept form the set `HEADINDEXES` of Listing A.1, while every other chunk position ranges over all the pieces. The restriction is exact, since it never removes the case $k^\star$ of the correctness argument above, and it shrinks the enumeration space from ℓ^{mR_B} to $(h\,\ell^{m-1})^{R_B}$, where h is the number of pieces kept for the head.

Reachability filter (proposed). The observation on the structure of B' above suggests a further refinement to the hybrid attack algorithm. Before committing to a full Gröbner basis computation for a case k, one could cheaply check whether the combination of subsets $(\mathcal{S}_{k_1}, \ldots, \mathcal{S}_{k_q})$ it selects is even reachable given the compression behavior above, and skip it if not.

Such a reachability filter would be independent of the probability ordering of Chapter 5, since the compression-based check removes cases that are provably unreachable, while the probabilistic ordering ranks the remaining ones. This means that these two criteria can be combined by applying the reachability filter first and then iterating over what remains in decreasing estimated-agreement order. However, neither this reachability filter nor the full $\mathbb{P}_{\text{trail}}$-sorted ordering is implemented in the enumeration actually run for the test results, which are left as future work.

5 Probabilistic Analysis

Chapter 4 turned the exact case-split by affine cover into a full attack by enumerating candidate cases and solving the resulting reduced system for each. Several questions were left open there, in particular:

1. which is the optimal ordering of the enumeration cases;
2. how many cases should we expect to try before finding one consistent with the true CICO solution;
3. how does this scale to real parameters of actual interest.

This chapter answers these by developing a probabilistic model of how likely a given case is to agree with the true lookup table on a uniformly random input. Then, we will use this model to reason about the enumeration order, the expected cost of the hybrid attack and how that cost is expected to scale. We derive the model from first principles, rather than starting from a simplifying independence assumption, because the natural first guess, that each chunk of a uniformly random $\mathbb{F}_p$-element is itself uniformly and independently distributed over $\{0, \ldots, 2^\omega - 1\}$, turns out to be false in general. Getting this right is what makes the resulting agreement-rate formula exact rather than heuristic, and, as we will discuss, it is also where the limits of this chapter's exactness lie once multiple Bar layers are chained together.

5.1 Bar Layer Approximation

Informally, we say that g is an *ε-approximation* ($0 < \varepsilon < 1$) of a function f over $\mathbb{F}_p$ if the two agree on an ε fraction of the inputs. This is the same notion of approximation used, in a different setting, by Monolith's designers to quantify how well an affine function tracks a prime-field nonlinear map [7, Lemma 4]. Here we adapt it to the chunk-wise structure of Skyscraper's Bar function, where the approximation is not a single affine map over $\mathbb{F}_p$ but an independent choice of affine map over $\mathbb{F}_p$ for each chunk, evaluated on chunk values in $\{0, \ldots, 2^\omega - 1\}$.

Note

This notion of approximation describes the Bar layer *as a whole* and is compatible with each individual chunk-level replacement being exact on its assumed subset, thus "exact" and "approximate" describe different things, namely, a single case versus the layer's behavior averaged over all inputs.

The overall approach is to approximate the polynomial system $\mathcal{F}$ modelling Skyscraper with its reduced version and then solve $\mathcal{F}'$ to obtain a candidate collision or pre-image. We fix a cover of ℓ candidate affine pieces $L_1, \ldots, L_\ell$ per chunk position and, for each BAR layer and for every chunk, is chosen a $L^{(j)} \in \{ L_1, \ldots, L_\ell \}$.

Definition: *ε-approximation of B'*

Let $B' : \mathbb{F}_p \to \mathbb{F}_p$ act on the base-D chunks $x = (z_0, \ldots, z_{m-1})$, with $D = 2^\omega$, through m parallel T-boxes $T : \mathbb{F}_{2^\omega} \to \mathbb{F}_{2^\omega}$. For each chunk z_j choose an affine piece $L^{(j)} : w \mapsto a^{(j)} \cdot w + b^{(j)}$ over $\mathbb{F}_{2^\omega}$, and let

$$\mathcal{S}_j \;=\; \{\, c \in \{0, \ldots, D-1\} : T(c) = L^{(j)}(c) \,\}$$

be the set of inputs on which $L^{(j)}$ agrees with the T-box. With $\hat{B}'$ we will denote the map obtained by replacing each T-box with its chosen affine piece $L^{(j)}$, one per chunk. Moreover, when a specific case k is in scope, we will make the choice explicit writing $\hat{B}'_k$ instead.
The *agreement count* Ω of the choice $(L^{(0)}, \ldots, L^{(m-1)})$ is the number of field elements on which $\hat{B}'$ and B' coincide, *i.e.*,

$$\Omega \;=\; \big|\{x = (z_0, \ldots, z_{m-1}) \in \mathbb{F}_p \,:\, \forall j \; z_j \in \mathcal{S}_j\}\big|,$$

and the *agreement rate* is $\varepsilon = \Omega / p$. We call $\hat{B}'$ an *ε-approximation* of B'.

Remark

Equivalently, ε is the probability that $\hat{B}'$ and B' coincide on a uniformly random input $\varepsilon = 1 - \delta$, where $\delta = \big|\{x \in \mathbb{F}_p : \hat{B}'(x) \neq B'(x)\}\big| / p$ is the relative Hamming distance between $\hat{B}'$ and B'. A larger ε thus means a stronger approximation.

Remark: *Chunk rotation*

The B' function applies a fixed cyclic rotation to the chunks before the T-box layer. This rotation merely relabels the input chunks, *i.e.*, the chosen subsets $\mathcal{S}_j$ and their associated $L^{(j)}$, and being a bijection on chunk positions, this does not affect Ω or ε. For clarity, we keep the index j associated with the chunk $X_{L,j}$ of the *input* to B', writing $\mathcal{S}_j$ for the subset constraining z_j rather than the more cumbersome index used in Section 4.4.

5.1.1 Field-Switch

We defined Ω as a count over $\mathbb{F}_p = \{0, \ldots, p-1\}$ and not over the full chunk product space $\{0, \ldots, D-1\}^m$ because it is exactly the point at which the naive per-chunk-uniform heuristic fails.

If chunks could be chosen freely and independently, the count would simply be the product $\prod_j |\mathcal{S}_j|$. But an element $x \in \mathbb{F}_p$ is only a valid input if its chunk decomposition $(z_0, \ldots, z_{m-1})$ is *strictly less than* p. Writing $(d_0, \ldots, d_{m-1})$ for the base-D digits of p, the naive product $\prod_j |\mathcal{S}_j|$ overcounts the tuples that decode to a value $\geq p$. Since it ignores the constraint entirely, it also gives no way to discard them correctly, because the chunks cannot be treated as independent once the field boundary is taken into account. In particular, as we will verify

concretely for $p = 587$ below, even the least significant chunk of a uniformly random $x \in \mathbb{F}_p$ is only approximately uniform over $\{0, \ldots, D-1\}$, and the most significant chunk can be very far from uniform. In the case of $p = 587$, it takes the value 2 with probability under 13% and never exceeds 2 at all, since $3 \cdot 16^2 > 587$.

We resolve this exactly by partitioning the inputs according to their *point of divergence* from p's digit sequence.

Definition: *Point of divergence*

Let $x \in \mathbb{F}_p$ have base-D chunk decomposition $(z_0, \ldots, z_{m-1})$ and let $(d_0, \ldots, d_{m-1})$ be the base-D digits of p. The *point of divergence* of x (with respect to p) is the smallest index $k \in \{0, \ldots, m-1\}$ such that $z_k \neq d_k$.

Since x is the canonical representative of an element of $\mathbb{F}_p$, *i.e.*, $0 \leq x < p$, such a k always exists and also $z_k < d_k$ holds at that index. This gives a partition of $\mathbb{F}_p$ into m disjoint classes, indexed by k, and we count the agreement condition separately within each class.

Proposition: *Exact agreement count*

The exact agreement count is represented by

$$\Omega = \sum_{k=0}^{m-1} \left(N_{\text{match}}^{(k)} \cdot N_{\text{break}}^{(k)} \cdot N_{\text{free}}^{(k)} \right),$$

where

- $N_{\text{match}}^{(k)}$ denotes the **matched prefix**, which ensures that all chunks prior to the divergence point k exactly match the modulo digits;
- $N_{\text{break}}^{(k)}$ is the **divergence step**, which counts, at the divergence point k, the choices that strictly guarantee $x < p$;
- $N_{\text{free}}^{(k)}$ is the **free suffix** and counts the unconstrained combinations for all remaining subsets, once the boundary condition has been satisfied at step k;

and their respective formulas are:

$$N_{\text{match}}^{(k)} = \begin{cases} 1 & \text{if } k = 0 \\ \prod_{j=0}^{k-1} \mathbb{1}(d_j \in \mathcal{S}_j) & \text{if } 1 \leq k \leq m-1 \end{cases},$$

$$N_{\text{break}}^{(k)} = \left| \{ y \in \mathcal{S}_k \mid y < d_k \} \right|,$$

$$N_{\text{free}}^{(k)} = \begin{cases} \prod_{j=k+1}^{m-1} |\mathcal{S}_j| & \text{if } 0 \leq k \leq m-2 \\ 1 & \text{if } k = m-1 \end{cases},$$

with $\mathbb{1}(\cdot)$ equal to 1 if its argument holds, 0 otherwise.

Proof

Every $x \in \{0, \ldots, p-1\}$ has a unique point of divergence k and, since $x < p$, $z_k < d_k$ at that index. Conversely, any digit vector with $z_j = d_j$ for $j < k$ and $z_k < d_k$ represents an integer smaller than p. Hence, the classes $D_k = \{x : z_j = d_j \; \forall j < k,\ z_k < d_k\}$, for $0 \le k \le m-1$, partition $\{0, \ldots, p-1\}$, and Ω is the sum over k of the number of $x \in D_k$ with $z_j \in \mathcal{S}_j$ for all j. Within D_k the conditions on the three groups of chunks concern disjoint indices and therefore multiply. The prefix is fixed to $(d_0, \ldots, d_{k-1})$ and agrees with the case if and only if $d_j \in \mathcal{S}_j$ for all $j < k$, which gives the factor $N^{(k)}_{\text{match}} \in \{0, 1\}$. The chunk z_k ranges over $\mathcal{S}_k \cap \{0, \ldots, d_k - 1\}$, which gives $N^{(k)}_{\text{break}}$. Each suffix chunk z_j, $j > k$, ranges freely over its own $\mathcal{S}_j$, since the divergence already guarantees $x < p$, which gives $N^{(k)}_{\text{free}} = \prod_{j>k} |\mathcal{S}_j|$ (an empty product for $k = m - 1$). Multiplying the three factors and summing over k gives the formula.

Because the field-boundary constraint $0 \le x < p$ is handled explicitly rather than approximated, Ω as given by the exact agreement count is an exact count, not a heuristic estimate.

Example: $m = 3$, $p = 587$

Consider $x \in \mathbb{F}_{587}$ split into $m = 3$ chunks of $\omega = 4$ bits, thus base $D = 16$. The digits of $p = 587$ are $(d_0, d_1, d_2) = (2, 4, 11)$. Suppose a chosen case constrains the chunks through

$$\mathcal{S}_0 = \{0, 1, 2, 4, 5\}, \qquad \mathcal{S}_1 = \{10, 11, 14\}, \qquad \mathcal{S}_2 = \{3, 9, 12, 15\}.$$

Applying exact agreement count

- **Divergence at k = 0**: we have that $N^{(0)}_{\text{match}} = 1$; $\mathcal{S}_0$ has two elements less than d_0, namely 0 and 1, so $N^{(0)}_{\text{break}} = 2$. Lastly, $N^{(0)}_{\text{free}} = |\mathcal{S}_1| \cdot |\mathcal{S}_2| = 3 \cdot 4 = 12$, thus the total contribution is $1 \cdot 2 \cdot 12 = 24$.
- **Divergence at k = 1**: since $d_0 = 2$ belongs to $\mathcal{S}_0$, we have $N^{(1)}_{\text{match}} = 1$. However, since every element of $\mathcal{S}_1$ is greater than d_1, we have that $N^{(1)}_{\text{break}} = 0$, so the final contribution is 0.
- **Divergence at k = 2**: we have that $N^{(2)}_{\text{match}}$ requires $d_0 \in \mathcal{S}_0$ plus $d_1 \in \mathcal{S}_1$. Since $4 \notin \mathcal{S}_1$, the indicator vanishes and thus the total contribution is 0.

Summing yields $\Omega = 24$ valid inputs out of 587, *i.e.*, an agreement rate of $\varepsilon = 24/587 \approx 4.09\%$.

5.1.2 Chaining Bar Layers

Exact agreement count gives an exact agreement rate $\varepsilon^{(r)} = \Omega^{(r)}/p$ for a *single* Bar layer case. The full Skyscraper permutation, however, applies R_B Bar layers ($R_B = 2$ for $R = 6$ rounds of the reduced variant, while $R_B = 4$ in the standard $R = 10$ variant). The hybrid attack therefore needs the probability that a full *trail* of cases, one per layer, agrees with the true permutation simultaneously across all of them.

A natural first model treats the R_B layers as independent and defines

$$\mathbb{P}_{\text{trail}} = \prod_{r=1}^{R_B} \varepsilon^{(r)}.$$

This independence is an assumption and not a consequence of the construction. The input of a later BAR layer is not an independently drawn field element but a deterministic function of the input of the first one. For two consecutive BAR rounds R_i and R_{i+1}, as rounds R_2 and R_3 in both parametrizations, the input of the second is $x_{L,i+1} = Y_i + \gamma_i + x_{R,i}$, *i.e.*, the output of the first BAR function plus the round constant and the additional state term $x_{R,i}$. When BAR rounds are separated by SQUARE rounds, as between R_3 and R_6 for $R = 10$, this value is further transformed by the SQUARE rounds in between. Whether the divergence-point partition of one layer is statistically independent of that of the next is therefore a claim about how these structured transformations interact with the partition; it is neither given by the construction nor tested in this thesis.

Therefore, $\mathbb{P}_{\text{trail}}$ is just an heuristic approximation, exact only in the degenerate case where the BAR layer inputs happen to be independent and uniform. This does not undermine its possible adoption in the hybrid attack since the enumeration order only needs $\mathbb{P}_{\text{trail}}$ to be a good *ranking* signal, because every candidate is verified exactly by the Gröbner basis computation itself. A trail whose true joint probability is somewhat higher or lower than $\mathbb{P}_{\text{trail}}$ predicts will simply be tried slightly earlier or later than optimal, at no cost for the correctness.

An exact treatment would need to track how the divergence-point partition of layer r propagates through the intermediate rounds and additions into the input distribution seen by layer $r+1$, rather than assuming that distribution is simply uniform again. This is what the input-output linkage heuristic outlined in Section 5.6 attempts to approximate.

5.2 Trail Selection

Granting $\mathbb{P}_{\text{trail}}$ as a ranking heuristic, we can now address the enumeration order for the hybrid attack, *i.e.*, we want to decide which trail, among all candidates, should be tried first.

A single BAR case chooses one $L^{(j)} \in \{ L_1, \ldots, L_\ell \}$ per chunk, so the space of single-BAR cases has size ℓ^m. Extending across the R_B BAR applications, an independent *choice* is made at each application, so the total number of trails is

$$\mathcal{N}_{\text{trails}} = (\ell^m)^{R_B} = \ell^{m \cdot R_B}.$$

The SQUARE rounds are not affected by the affine cover representation, so they do not contribute with any factor.

The agreement rate of any individual trail is very low, especially as p and m grow, so the goal is not a single high-probability trail, instead identifying the *dominant* trails, *i.e.*, those whose rate is well above the average. The hybrid attack should evaluate $\mathbb{P}_{\text{trail}}$ for the candidate trails and try them in decreasing order while also take into consideration the following criteria:

1. if $|\mathcal{S}_j|$ is large, maximizing the degrees of freedom N_{free};
2. if $\mathcal{S}_j$ contains integers strictly below the bound digit d_j, then we need N_{break} to be nonzero at the divergence point.

For the most significant chunk, a case whose set contains several low integers (*e.g.*, $\mathcal{S}_1 = \{0, 1, 2, 4, 5\}$) easily satisfies the N_{break} condition, keeping the corresponding branch of the Ω partition alive. Conversely, a set of only high integers (*e.g.*, $\mathcal{S}_3 = \{10, 11, 14\}$) at the most significant chunk drives that branch to zero.

Table 5.1: Top-10 trails ranked by agreement rate for $m = 3$, $R_B = 2$, $\ell = 5$, $p = 587$ and $\omega = 4$.

B'_1	B'_2	$\mathbb{P}_{\text{trail}}$
(L_1, L_1, L_1)	(L_1, L_1, L_1)	0.014221
(L_1, L_1, L_1)	(L_1, L_2, L_1)	0.010970
(L_1, L_2, L_1)	(L_1, L_1, L_1)	0.010970
(L_1, L_1, L_1)	(L_2, L_1, L_1)	0.009142
(L_2, L_1, L_1)	(L_1, L_1, L_1)	0.009142
(L_1, L_2, L_1)	(L_1, L_2, L_1)	0.008463
(L_1, L_1, L_1)	(L_1, L_3, L_1)	0.008126
(L_1, L_3, L_1)	(L_1, L_1, L_1)	0.008126
(L_1, L_1, L_1)	(L_2, L_2, L_1)	0.007313
(L_2, L_2, L_1)	(L_1, L_1, L_1)	0.007313

Out of $\ell^{m \cdot R_B} = 15625$ candidate trails, only 625 have nonzero agreement rate. For instance, pairing (L_1, L_1, L_1) on B'_1 with (L_1, L_3, L_3) or (L_1, L_1, L_5) on B'_2 both give $\mathbb{P}_{\text{trail}} = 0$, since in each case the choice on B'_2 violates the strict modulo bound at its most significant diverging chunk, forcing $N_{\text{break}} = 0$ for every divergence class. Such trails can be discarded without any Gröbner basis computation at all, simply by evaluating exact agreement count on each layer independently, which is far cheaper than a Gröbner basis computation.

5.2.1 Clustering Phenomenon

Beyond the majority of the candidate trails evaluating to zero, the top of the ranking is not spread evenly across the 625 nonzero trails. This is expected, as some single-BAR cases recur disproportionately often among the highest-ranked trails, appearing paired with many choices at the other BAR layer rather than only with themselves. Extending the previous table to the top 50 trails and counting, for each single-BAR case, how often the case (L_1, L_1, L_1) appears at either layer among them, resulted in 24 cases of 100 layer-appearances. The next few most frequent cases ((L_1, L_1, L_2), (L_1, L_2, L_1), (L_1, L_1, L_3), (L_1, L_2, L_2)) are all cases that, like (L_1, L_1, L_1), place L_1 on the most significant chunk.

This clustering behavior is a direct consequence of the B' compression explained in Section 4.6. Moreover, $N^{(0)}_{\text{break}}$ is nonzero only when $\mathcal{S}_0$ contains a value below d_0 and, once satisfied, contributes a factor proportional to $|\mathcal{S}_0|$ and to every subsequent chunk's cardinality via $N^{(0)}_{\text{free}}$. A case that places a large low-valued piece like L_1 at the most significant chunk, tends to *dominate* simultaneously across many different trails. This gives a concrete, checkable criterion for selecting effective cases, beyond simply sorting all ℓ^m of them by their computed rate, which consists of preferentially place the cover's largest, lowest-valued piece at the most significant chunk of every BAR layer first.

5.3 Parameter Dependence

The agreement rate depends strongly on the structure of the prime p since it determines how favorable a given Skyscraper parametrization is to the attack of Chapter 4 before any Gröbner basis computation is attempted at all.

Within the B' function the T-box operates on $\mathbb{F}_{2^\omega}$, so the number of chunks m is not a free parameter, instead is fixed jointly by ω and p. Recall that $m = \lceil \log_{2^\omega}(p) \rceil$, *e.g.*, for $\omega = 4$ and $p = 587$, this gives $m = 3$. This constraint underlies the parameter choices of Table 3.2.

Fixing $\omega = 4$ and the five cases $L_1, \ldots, L_5$ of Figure 4.2, the governing quantity is the base-D representation $(d_0, \ldots, d_{m-1})$ of p, in particular its leading digit d_0. By exact agreement count, the branch $k = 0$ contributes to Ω as soon as $\mathcal{S}_0$ contains an element below d_0. Conversely, if $\mathcal{S}_0$ contains no element $\leq d_0$, both the $k = 0$ branch and the matching prefix of every deeper branch fail (since $N^{(k)}_{\text{match}}$ for $k \geq 1$ requires $d_0 \in \mathcal{S}_0$), forcing $\Omega = 0$ entirely. The most significant chunk therefore acts as a *gatekeeper* for the whole agreement count, which is exactly the structural fact behind the clustering phenomenon.

Since the number of viable trails is dictated by d_0, the extreme regimes occur at the chunk-count thresholds D^t where m increments:

- immediately *below* such a threshold, p's base-D representation is $(D-1, D-1, \ldots)$, so $d_0 = D - 1$ is maximal and the gatekeeper is loosest;
- immediately *above* it, p acquires a new leading chunk equal to 1 (so m increases by one) and $d_0 = 1$ is minimal, making the gatekeeper tightest.

Crossing a single threshold therefore simultaneously adds a chunk and resets d_0 from maximum to minimum, so two primes lying just below and just above the same threshold cover the range of possible gatekeeper behavior.

The following Table 5.2 reports the number of single-Bar cases with nonzero agreement rate $\mathcal{N}_{\neq 0}$ out of ℓ^m and the dominant single-Bar rate $\varepsilon^\star = \Omega_{\max}/p$, dominant trail $(L_1, \ldots, L_1)$.

Table 5.2: Dependence of the agreement rate on the prime structure, for $\omega = 4$ and $\ell = 5$.

p	m	$(d_0, \ldots, d_{m-1})$	$\frac{\mathcal{N}_{\neq 0}}{\ell^m}$	$\frac{\mathcal{N}^2_{\neq 0}}{\ell^{2m}}$	$\varepsilon^\star$
251	2	(15, 11)	25/25	625/625	0.099602
65521	4	(15, 15, 15, 1)	625/625	390625/390625	0.009539
$2^{20} - 3$	5	(15, 15, 15, 15, 13)	3125/3125	9765625/9765625	0.002980
$127 = 2^7 - 1$	2	(7, 15)	15/25	225/625	0.196850
2953	3	(11, 8, 9)	125/125	15625/15625	0.042330
257	3	(1, 0, 1)	25/125	625/15625	0.101167
587	3	(2, 4, 11)	25/125	625/15625	0.119250
65537	5	(1, 0, 0, 0, 1)	625/3125	390625/9765625	0.009552

Table 5.2 is divided into three sections:

- top: is the case of full space when d_0 is large;

- middle: intermediate for moderate d_0;
- bottom: ℓ^{m-1} when $d_0 = 1$;

and the two representative cases that illustrate the table are:

- **Below a threshold**, for instance $p = 251$ ($m = 2$) and $p = 65521$ ($m = 4$), just below 2^8 and 2^{16}. Their base-16 digits, $(15, 11)$ and $(15, 15, 15, 1)$, have maximal $d_0 = 15$, so every subset satisfies the divergence condition and all ℓ^m single-Bar cases are viable (25/25 and 625/625).
- **Above a threshold**, for example $p = 257$ ($m = 3$) and $p = 65537$ ($m = 5$), just above the same thresholds. The extra leading chunk is minimal, $d_0 = 1$ in both cases, so only L_1 can satisfy the gatekeeper condition, forcing L_1 on the most significant chunk and leaving exactly ℓ^{m-1} of the ℓ^m cases viable, a fraction $1/\ell = 20\%$.

5.4 Scaling to Real Parameters

The analysis so far has used $\omega = 4$ throughout, since it is small enough to inspect, verify and run full experiments on. However, the value of actual cryptanalytic interest is $\omega = 8$, together with p at or near Skyscraper's proposed security parameters ($2^{248} < p < 2^{256}$, so that $m = \lceil \log_{2^8} p \rceil = 32$) and the standard permutation of $R = 10$ rounds, with $R_B = 4$ Bar layers.

As shown in Figure 4.1, we also computed an affine cover for the $\omega = 8$ lookup table with the cover selection heuristics of Section 4.2, and it is listed in Appendix B. The cover is composed of $\ell_8 = 42$ pieces that jointly cover all 256 possible T inputs, but with some pieces overlapping (this was not the case for $\omega = 4$). Because no algorithm for the cover select has been yet implemented, such cover is greedy and there is no assurance whatsoever that is the optimal. The largest piece covers $|L_{\max}| = 34$ of the $D = 2^8 = 256$ inputs, a fraction $34/256 \approx 13.3\%$, which is smaller than L_1 for the $\omega = 4$ case which had $5/16 = 31.25\%$, but is still a good value for the dominant piece.

Therefore, we estimate how the cost of the attack is expected to scale on a real instance in two ways.

- We can provide an estimation based on the **cover size** ℓ. The bound for the worst-case is $\mathcal{N}_{\neq 0} \leq \ell_8^m$ per Bar layer, hence is $\mathcal{N}_{\neq 0}^{R_B} \leq \ell_8^{mR_B}$ overall. Using the values of our proposed cover for $\omega = 8$, we have that $\ell_8 = 42$, $m = 32$, and $R_B = 4$, thus the exponent is $mR_B = 128$ and the worst-case bound becomes $42^{128} \approx 10^{208}$.
- Another estimation can be made from the **per-chunk odds**. If each of the $mR_B = 128$ chunk positions is treated as independently uniform, the probability that all of them land in the dominant piece $\mathcal{S}_{\max}$, *i.e.*, the probability of the most likely trail, is $\left(|\mathcal{S}_{\max}| / 2^\omega\right)^{mR_B}$. Any deterministic enumeration order succeeds within its first t trials with probability at most t times this value, so the expected number of trials is at least about half of its inverse, $\frac{1}{2}\left(2^\omega / |\mathcal{S}_{\max}|\right)^{mR_B}$. Substituting the real $|\mathcal{S}_{\max}| = 34$, this is about $(\frac{256}{34})^{128} \approx 10^{112}$ up to a factor 2, and it is a heuristic estimate, since the independence assumption is only approximate.

Both estimates, despite having a huge difference in the magnitude, they still agree on the fact that the trial count is *astronomically large* and well beyond Skyscraper's own claimed 124-bit ($\approx 2^{124} \approx 10^{37}$) security level, which is *far from a practical computation*.

The proposed hybrid attack therefore does not threaten Skyscraper at its real parameters, although there may be room for narrowing the gap with a threatening attack. This of course implies that most of the work left for the future is needed before reaching such a state, which include deeper analyses on low-level Skyscraper structure, to achieve effective heuristics that could improve drastically the current state of the attack.

5.5 Attack Complexity

We distinguish two complexity questions:

1. the complexity of the *attack*, *i.e.*, how expensive is to run;
2. the complexity of the *probabilistic model* itself, *i.e.*, how expensive is it to compute the probabilities needed to decide what order to try the affine pieces.

Moreover, the first complexity is dependent on the second, thus we need to answer to both in order to have a good complexity bound.

Attack complexity. Recalling the cost model of Chapter 4, the attack's total cost is the number of cases tried times the cost of one Gröbner basis computation. The number of cases tried is bounded above by $\mathcal{N}_{\neq 0}$, and, per Section 5.7 below, is expected in the typical case to be far smaller, governed by the rank of the true trail under the ordering of Section 5.2. The cost of one Gröbner basis computation on the reduced system is bounded via the Macaulay-bound in terms of m, ω and the sizes of the assumed subsets $\mathcal{S}_{k,j}$.

Model computation complexity. Evaluating exact agreement count for a single case and a single Bar layer, requires computing $N^{(k)}_{\text{match}}$, $N^{(k)}_{\text{break}}$ and $N^{(k)}_{\text{free}}$ for each of the m divergence classes k. Each term is computable in $\mathcal{O}(m)$ time given the subsets' sizes and a sorted representation of each $\mathcal{S}_j$ (to count elements below d_k via binary search), so a single evaluation of Ω costs $\mathcal{O}(m^2)$ in the worst case, entirely negligible next to any Gröbner basis computation. Ranking all ℓ^m single-Bar cases by brute force costs $\mathcal{O}(\ell^m \cdot m^2)$, which is the same order as the search space itself and is the dominant cost since for $\omega = 8$ (we have $\ell_8^m = 42^{32}$, with respect to our cover). This is exactly why pruning criteria of trail selection are valuable independently of their effect on the attack's own cost. By cheaply discarding trails with zero agreement rate, it keeps the cost of computing the ranking from becoming a bottleneck, and thus avoids the need to evaluate the full agreement count on the entire case space.

5.6 Cross-Layer Heuristics

This section sketches a refinement that is neither implemented nor evaluated in this thesis. Bar layer chaining identified the propagation of the divergence-point distribution through the intervening Square rounds (Figure 5.1) as the missing ingredient for an exact cross-layer probability model.

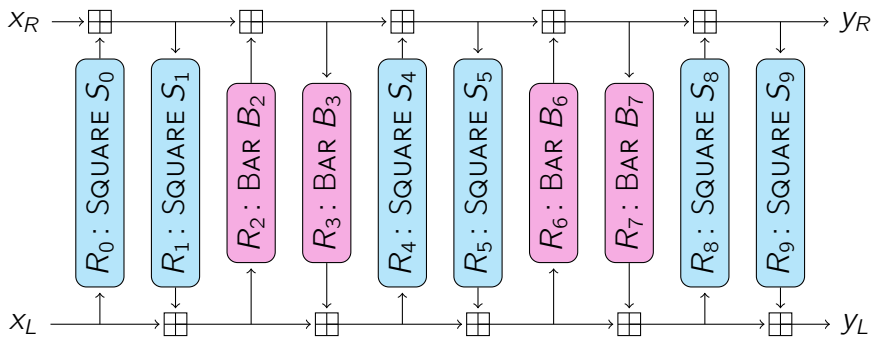


Figure 5.1: Full-round Skyscraper permutation for $R = 10$ rounds. With respect to Figure 3.2, here the focus is on the Feistel-like structure and to how the value of left and right branches are propagated between round layers (cross-layer heuristic of Section 5.6).

Given a case $(L^{(0)}, \ldots, L^{(m-1)})$ for the first BAR layer and the known SQUARE-round transformation between the two layers, the set of possible inputs to the second BAR layer, conditioned on the first layer's case having been correct, is the image of $\mathcal{S}_0 \times \cdots \times \mathcal{S}_{m-1}$ under the SQUARE-round map. Rather than assuming, as $\mathbb{P}_{\text{trail}}$ does, that this image is uniformly distributed over $\mathbb{F}_p$, an input-output linkage estimate would compute (or approximate) the distribution of the chunk decomposition of this image directly. It would then use the exact agreement count against that distribution to rank second-layer cases conditionally on the first layer's choice. This would replace the unconditional ranking of trail selection (layer's cases ranked independently) with a conditional one, in which the second layer's ranking depends on which case was assumed for the first.

Concretely, this estimate can be computed without any additional cost, because every quantity involved is fixed and public. The SQUARE-round constants and the affine cover do not depend on the CICO input, so nothing about them needs to be guessed or solved for. Rather than ranking every trail across all R_B BAR layers at once, this suggests the following two-step filtering scheme:

1. after fixing a case for the first BAR layer, compute the set of possible B'-compressed outputs it can produce (*i.e.*, using only the *input*-side subsets $\mathcal{S}_j$ fixed so far, as done by $\mathbb{P}_{\text{trail}}$);

2. propagate this set through the intervening SQUARE round to obtain the possible *inputs* to the second BAR layer, and check which second-layer cases are even consistent with this reduced set, before enumerating over them.

Because the second step only needs to check membership against an already known set, rather than recomputing an agreement count from scratch for every BAR chaining combination, this cross-layer check adds only a bounded enumeration cost. It nevertheless allows the pruning of combinations across layers that $\mathbb{P}_{\text{trail}}$ cannot detect. Making this estimate concrete and evaluating its effect on the number of iterations hybrid attack actually needs is left as future work, as detailed in Section 7.3.

5.7 Expected Search Cost

We can now return to the question that opened this chapter, *i.e.*, how many iterations of the attack are expected before finding a consistent trail, and whether the search overhead remains worthwhile against the per-iteration savings of the reduced system.

If the attack tries trails in strictly decreasing order of $\mathbb{P}_{\text{trail}}$, and if the real trail is itself drawn according to this same distribution, then the expected number of iterations before success is the expected *rank* of the real trail under this ordering. Two regimes bound this quantity, namely if the top-ranked trail is already the correct one, the expected number of iterations is 1. On the other hand, the worst case is when every one of the $\mathcal{N}_{\neq 0}$ nonzero trails might need to be tried, giving $\mathcal{N}_{\neq 0}$ as an upper bound, as analyzed in Section 5.4 for a real scenario.

This rank-based estimate inherits the same approximation as $\mathbb{P}_{\text{trail}}$ itself because it treats the real trail as drawn from the distribution of $\mathbb{P}_{\text{trail}}$, so refining it further is tied to the cross-layer heuristics of Section 5.6.

Formally, if the trails are tried in an order π and the true trail t is drawn with probability $\mathbb{P}_{\text{trail}}(t)$, suitably normalized over the trails with nonzero agreement rate, the expected number of iterations is

$$\mathbb{E}[\pi(t)] = \sum_{t} \mathbb{P}_{\text{trail}}(t)\, \pi(t),$$

which lies between 1 and $\mathcal{N}_{\neq 0}$. Two caveats apply when comparing this quantity with the combination ratios of Table 6.4. First, it refers to the position of the true trail, whereas the attack stops at the first trail whose reduced system yields a verified solution, and a digest in general has more than one pre-image. Second, the enumeration evaluated experimentally is not ordered by $\mathbb{P}_{\text{trail}}$, but is lexicographic over the space left by the head restriction. A quantitative comparison between the model and the experimental ratios is not carried out in this thesis and is left as future work.

6 Experimental Results

This chapter reports an experimental evaluation of the hybrid attack explained in Chapter 4 against a parameter-reduced Skyscraper instance, following the same values used in the original paper for Skyscraper Gröbner basis cryptanalysis, in order to make these directly comparable. We analyze not only the running times but also how much of the enumeration space the attack needs to explore before succeeding. We discuss both in light of the probabilistic analysis of the previous chapter, without a quantitative comparison with the predictions of the model, which is left as future work.

6.1 Experimental Setup

We implemented the hybrid case-split attack in SageMath and evaluated it on the parameter-reduced instance of $\omega = 4$, for both $R = 6$ and $R = 10$ designs.

Every timing reported in this chapter refers to the time needed by the *entire* attack of Section 4.5 (Listing A.4) to terminate with a solution. Thus, these times do *not* refer to the time of a single Gröbner basis computation, but they include every case tried before the successful one, each requiring its own reduced-system Gröbner basis computation, plus the cost of generating and iterating the enumeration order itself. This distinction is needed to interpret the values presented below correctly.

Skyscraper's own reported timings in Table 3.2 were obtained by its designers on their own hardware and software setup [8, Section 5]. Since the machine on which we ran our experiments is *different* from theirs, we could not achieve a state-of-the-art benchmark comparison, so our presented data should be read as *indicative* with respect to their specifications.

Table 6.1: Hardware and Software specifications of the test environment.

	Designers' system	Our system
CPU	AMD Ryzen 9 7900X	Intel i7-8550U
GPU	—	AMD Radeon 530 4 GB GDDR3
RAM	128 GB	8 GB DDR4
OS	Linux	Void Linux
Kernel	—	6.18.48
Environment	—	SageMath 10.9
Backend	—	Singular
Solving algorithm	—	slimgb

Everything has been run single-threaded, even the enumeration, avoiding any possible parallelization, in order to keep the comparison fair.

Instances and constants. The round constants are $\gamma_0 = \gamma_{R-1} = 0$. For the remaining rounds, γ_i is obtained by hashing with SHA-256 the string formed by the round index (4 bytes, big-endian), the ASCII label `Skyscraper` and zero padding up to 28 bytes, then reading the digest as a big-endian integer and reducing it modulo p. The Montgomery constant is $\sigma = 2^{64\lceil\lceil\log_2 p\rceil/64\rceil} \bmod p$, *i.e.*, $2^{64} \bmod p$ for all the primes tested. Each pre-image instance is generated by drawing $x_R \in \mathbb{F}_p$, setting $x_L = 0$ and taking as target digest the right output of $\mathcal{P}(0, x_R)$, so that every instance has at least one solution. For $p \leq 191$ every $x_R \in \mathbb{F}_p$ is used exactly once, while for larger primes a sample of values is drawn uniformly at random (300 for the instances of Table 6.4), without a fixed seed. The reported hybrid times and combination ratios are average means over the independent searches performed for each instance. In particular, searches that exhausted their enumeration without a verified solution are reported separately as a failure, but during our tests this never happened for any prime evaluated. The Gröbner basis is computed through SageMath with Singular's `slimgb`, without any timeout. The source code is available at [34] and the affine covers used are listed in Appendix B.

6.2 Average Running Time

The following table reports the running times for both the standard Gröbner basis attack (already presented in Table 3.2) and our hybrid attack based on the affine cover. The tests have been run for $\omega = 4$ and both $R = 6$ and $R = 10$ cases.

Table 6.2: Skyscraper-CICO pre-image search solving times for $\omega = 4$. Our times are means over the independent searches of Section 6.1. The designers' times are those reported in [8, Table 16] for $R = 6$, measured on different hardware and software: the two groups of columns are not a same-machine benchmark.

p	m	Skyscraper		Our results	
		$R = 6$ (s)	$R = 10$ (s)	$R = 6$ (s)	$R = 10$ (s)
37	2	1	—	0.07	2
109	2	2	—	0.5	56
163	2	3	—	1.2	513
191	2	4	—	1.8	1264
587	3	243	—	4	63240
1237	3	7 550	—	25	—
1361	3	18 974	—	19	—
1399	3	18 912	—	20	—
2297	3	51 103	—	132	—
2953	3	123 706	—	177	—
4099	4	—	—	6217	—
10973	4	—	—	45991	—

For a fair comparison, we report "solving times", since in the authors' methodology, computing a single Gröbner basis meant solving the Skyscraper-CICO problem, while in our case it meant running the entire attack, *i.e.*, enumeration of each trail and Gröbner basis computation for each. Table 6.4 below describes the same $p = 587$, $R = 6$ instance of Table 6.2 from a different angle, and the two tables are best read together. For $p = 587$ ($m = 3$, $R_B = 2$ Bar layers) the head restriction keeps a single piece for the most significant chunk, since $d_0 = 2$ and $\{0, 1, 2\} \subseteq \mathcal{S}_1$, so the enumeration space has $(1 \cdot 5^2)^2 = 625$ cases instead of the $\ell^{mR_B} = 5^6 = 15625$ of the unrestricted one. The mean combination ratio of 23.4% is measured over these 625 cases, *i.e.*, roughly 146 cases tried on average before success. Dividing the 4 s mean solving time by 146 gives an average cost of about 27 ms per case, which includes the reduced-system Gröbner basis computation and the overhead of the enumeration. Since the implemented enumeration order only applies the head restriction of Section 4.6, every one of these attempts corresponds to a genuine Gröbner basis computation on the reduced system, thus this running time could plausibly be lowered by implementing the strategies of Chapter 5.

The original paper reports no results for $R = 10$ nor for $m = 4$, while Table 6.2 reports hybrid solving times for $m \leq 3$ at $R = 10$ and for $m = 4$ at $R = 6$.

Finally, Table 6.3 makes the comparison explicit by computing the speedup factor for every designers' entry of Table 6.2.

Table 6.3: Speedup of the hybrid attack over the baseline timings of [8], which were measured on different hardware and are therefore indicative.

p	m	Speedup
37	2	14.3
109	2	4.0
163	2	2.5
191	2	2.2
587	3	60.8
1237	3	302.0
1361	3	998.6
1399	3	945.6
2297	3	387.1
2953	3	698.9

Two observations follow directly from the above table:

- **Speedup jumps sharply with m.** Every $m = 3$ instance achieves a speedup at least an order of magnitude larger than any $m = 2$ instance (60.8 at minimum, versus 14.3 at maximum for $m = 2$). This is consistent with Section 4.4 since increasing m increases the number of chunk positions whose degree the affine case-split collapses, so the saving compounds multiplicatively across chunks, while the baseline's cost grows correspondingly faster with m.

- **Speedup is not monotonic in p, even within a fixed m.** the speedup is not monotonic but can even *decrease* as for $m = 2$ ($14.3 \to 4.0 \to 2.5 \to 2.2$). This is the case since the speedup depends both on p and m together, in particular, is governed by the base-D

digit structure of p and the m, because the leading digit d_0 drives $\mathcal{N}_{\neq 0}$ and hence how many trials the attack needs.

Relative to the designers' reported timings, which were obtained on different hardware and software and are therefore indicative only, the overall range across every tested instance is best summarized as "roughly 2× to close to three orders of magnitude". This is the figure used consistently in Section 1.3 and Chapter 7.

For $R = 10$, the original paper reports no results at all, while the hybrid attack solves the $m = 2$ instances with $p \leq 191$ and the $m = 3$ instance $p = 587$ within the timings of Table 6.2, whose growth with p and m is steep.

6.3 Combination Ratio

Choosing an effective enumeration order for the case-split is central to the hybrid attack's performance, since the number of iterations actually needed determines the total running time. Table 6.4 reports, for a pre-image search under this ordering, the ratio between the number of cases *tried* before success and the total number of cases of the enumeration space. A case is *tried* when the attack substitutes it into the reduced system and attempts to solve it. The enumeration space is the space left by the head restriction of Section 4.6, whose size is $(h\,\ell^{m-1})^{R_B}$, with h the number of pieces kept for the most significant chunk (`comb_size` in Listing A.3). It equals ℓ^{mR_B} only if no piece is discarded. Each row's "# searches" column gives the number of independent pre-image searches (each against a different target digest, obtained as described in Section 6.1, for the corresponding p) that the mean and threshold percentages in that row are computed over. The two threshold columns "$\leq 10\%$" and "$\leq 20\%$", report the percentage of those independent searches that succeeded respectively within the first 10% and 20% of the exhaustive search space.

The resulting distribution of per-search combination ratios for a subset of these results, for $R = 6$, is plotted as a histogram in Figure 6.1.

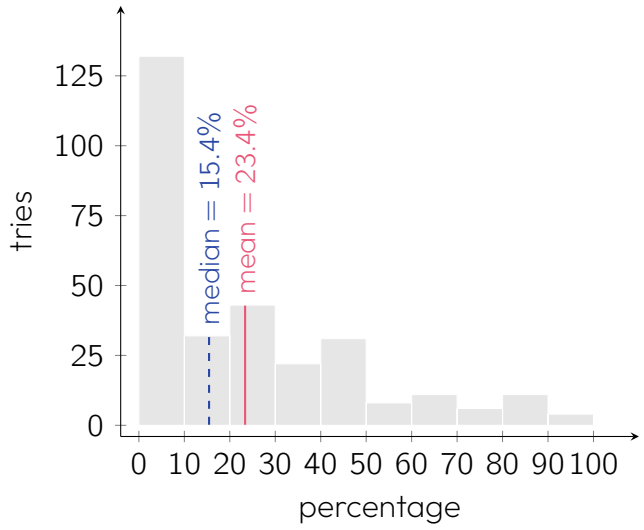


Figure 6.1: Distribution of the per-search combination ratio (percentage of the exhaustive search space tried before success) for the $p = 587$, $R = 6$ row of Table 6.4.

Table 6.4: Skyscraper-CICO probabilistic attack results, highlighting the ratio between the number of tried combinations to successfully recover the pre-image and the size of the enumeration space after the head restriction, $(h\ell^{m-1})^{R_B}$.

p	m	rounds R	# searches	comb. tried/comb. max		
				mean (%)	$\leq 10\%$ (%)	$\leq 20\%$ (%)
37	2	6	37	18.5	40.5	59.5
		10		21.2	48.6	54.0
109	2	6	109	32.4	33.9	53.2
		10		31.1	28.4	45.9
163	2	6	163	21.9	49.1	63.2
		10		19.2	49.7	67.5
191	2	6	191	21.3	51.8	63.9
		10		20.7	52.1	62.6
587	3	6	300	23.4	42.7	54.0
1237	3	6	300	20.2	38.0	60.0
1361	3	6	300	19.1	42.3	65.7

Two observations follow from Table 6.4 beyond the raw figures.

1. measured over the enumeration space left by the head restriction, the mean combination ratio stays under 35% across every parameter setting tested, and in most cases at least half of the individual searches succeed within the first 20% of that space. This space is itself much smaller than the unrestricted one (*e.g.*, 625 instead of 15625 cases for $p = 587$, $R = 6$), so the number of cases actually tried is a small fraction of ℓ^{mR_B}. The experiments, however, do not separate the contribution of the head restriction from that of the cheaper reduced-system Gröbner basis computation. Implementing the full sort of the enumerations by $\mathbb{P}_{\text{trail}}$ could lower these ratios further, but this has not been tested.

2. the ratio does not shrink monotonically as p grows within a fixed m, mirroring the non-monotonic pattern already noted for the raw speedups in Section 6.2. This is consistent with the same underlying cause, namely that the combination ratio depends on p's specific digit structure via $\mathcal{N}_{\neq 0}$ rather than on the magnitude of p; with only a few primes per value of m, however, this explanation is not tested quantitatively.

7 Conclusion

7.1 Summary

This thesis investigated whether the interpolation polynomial is the only faithful algebraic representation of a lookup-table S-box that can be given to a Gröbner basis solver, and whether the field-switching bottleneck it introduces can be avoided. Using Skyscraper as a case study, we showed that the bottleneck cannot be removed entirely, but that it can be reduced by choosing a different representation of the lookup table, at least on the parameter-reduced instances considered here.

The representation we proposed is an *affine cover* of the lookup table T. Viewed as a set of points in $\mathbb{F}_p^2$, T admits a small number of affine pieces, each of which agrees with it exactly on a subset of its domain. Under the assumption that the input of a lookup lies in the subset of a given piece, replacing T by that piece lowers the degree of the recomposition equation from $2^\omega - 1$ to 1. Since the correct piece is not known in advance, we turned this observation into a hybrid attack that enumerates the candidate cases and solves the reduced system of each case with a Gröbner basis computation. Every extracted solution is then verified against the permutation, so that no incorrect solution can be returned.

The cost of the attack depends mainly on how many cases must be tried, and not only on the cost of each trial. For this reason, we developed a probabilistic model of the agreement rate of a case, derived from the base-2^ω digits of p instead of the heuristic "each chunk is independently uniform", which is measurably false once the field-boundary constraint $0 \leq x < p$ is taken into account. The model allowed us to rank trails of cases, to characterize the clustering of the most likely trails, to describe how the agreement rates depend on the arithmetic structure of p and to bound how the number of cases scales to the real parameters of Skyscraper. It predicts agreement rates of trails, but its expected search cost has not been compared quantitatively with the experimental combination ratios.

Finally, we evaluated the attack on the parameter-reduced Skyscraper instances used in the analysis of the design paper. With respect to the timings reported by the designers, which were measured on different hardware and are therefore indicative, we obtained speedups from roughly two times to almost three orders of magnitude depending on the instance. We also solved the settings $m = 3, R = 10$ and $m = 4, R = 6$, which the original evaluation did not report. Within the enumeration space left by the head restriction, the fraction of cases explored before success stayed under 35% on average across the tested instances, and in most settings at least half of the individual searches succeeded within the first 20% of that space. The experiments do not separate the gain due to the smaller enumeration space from the gain due to the cheaper individual Gröbner basis computations.

7.2 Limitations

The attack has several limitations, which are discussed in Chapter 5 and in particular in Section 5.4.

Parameter scale. Every experiment was conducted on parameter-reduced instances with $\omega = 4$ and $p \leq 10973$, and could not be extended to the proposed parameters of Skyscraper, *i.e.*, $p \geq 2^{248}$ and $\omega = 8$.

Affine cover selection. The cover used for $\omega = 4$ was found by direct inspection of the scatter plot of T, while the cover for $\omega = 8$ was found with a greedy algorithm following the criteria of Section 4.2. No algorithm for finding an optimal cover for an arbitrary ω is available, so the optimality of both covers is not guaranteed, although the whole attack depends on this choice.

Enumeration order only partially implemented. Chapter 5 develops $\mathbb{P}_{\text{trail}}$ as a ranking of the trails, and Section 4.6 describes a reachability filter based on the compression behavior of B'. The implementation evaluated in Chapter 6 applies only the head restriction on the most significant chunk of each Bar layer, and implements neither the $\mathbb{P}_{\text{trail}}$-sorted order nor the reachability filter. The reported speedups and combination ratios therefore do not reflect the full enumeration order, whose benefit has not been measured.

Exponential cost in the number of chunks. The number of cases to try grows as ℓ^{mR_B}, *i.e.*, exponentially in both the number of chunks m of each Bar application and the number of Bar layers R_B. At the real parameters ($\omega = 8$, $m = 32$ and $R_B = 4$), this exponent is large enough that, as computed in Section 5.4, even optimistic assumptions on the quality of the cover leave the worst-case number of cases far beyond the 124-bit security level claimed by the designers.

7.3 Future Work

We separate the future directions into improvements of the attack and of its analysis, which follow from the open questions identified in this thesis, and broader directions that apply the same technique beyond Skyscraper.

7.3.1 Attack Improvements

Implementing the full enumeration order. As detailed in Section 4.5, the evaluated implementation applies only the head restriction. Implementing the $\mathbb{P}_{\text{trail}}$-sorted order and the B' reachability filter, individually and combined, and measuring again the running times and combination ratios of Chapter 6 is the most direct follow-up, since both components are already specified and mainly require engineering effort.

A rigorous cross-layer probability model. Treating the agreement across Bar layers as independent is a heuristic whose error is not quantified in this thesis. A rigorous treatment would track how the divergence-point partition of the output distribution of one Bar layer

propagates through the intermediate rounds into the input distribution of the next one, and would replace the product formula $\mathbb{P}_{\text{trail}}$ with a more accurate one. The input-output heuristic of Section 5.6, with its two-step filtering scheme based on the public SQUARE-round constants and affine cover, is a starting point. It would also allow the expected-rank calculation of Section 5.7 to be made precise and validated against the combination ratios of Section 6.3.

A theoretical complexity bound for the reduced system. We observed, both analytically and experimentally, that the Gröbner basis complexity of the case-split system is much lower than that of the interpolation-based system. A theoretical statement of this gap is still missing, *e.g.*, a bound on the degree of regularity or on the solving degree of the reduced system that does not rely on the generic Macaulay bound. Such a bound would replace the extrapolation of Section 5.4 with a rigorously derived statement, instead of the broad estimate given there.

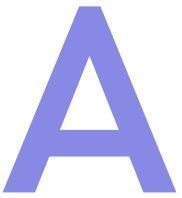
An algorithm for optimal affine covers. The covers for $\omega = 4$ and $\omega = 8$ were not checked for optimality, and the second one was computed greedily following the minimum-pieces and minimum-overlap criteria of Section 4.2. For arbitrary ω, which may appear in other lookup-table-based AO primitives, a dedicated covering algorithm is needed, and it would also change which optimizations based on the clustering phenomenon and on the cross-layer heuristics are possible.

Optimizations for the SQUARE layer. This thesis focused on the BAR layer because it is the source of the field-switching bottleneck, but the SQUARE rounds still contribute to the polynomial system. Gröbner basis optimizations that exploit the structure of the squaring polynomial could therefore accelerate the attack further, independently of the improvements made for the BAR layer.

7.3.2 Extension to Other Primitives

As discussed in Chapter 1, the case-split of Chapter 4 is not specific to Skyscraper, since any AO primitive that uses a lookup table in a mixed-field design is a potential target, for instance `ReinforcedConcrete` and `Monolith`. Adapting the hybrid attack and its probabilistic analysis to these designs is a natural next step, and it would test how general the technique is, since each of these primitives uses different S-boxes, chunk sizes and field-switching mechanisms.

Implementations



This appendix reports the SageMath source excerpts from [34] that implement the constructions described in Chapter 4 and which have been used to collect the results of Chapter 6.

A.1 Affine Cover

Listing A.1 defines:

- the $\omega = 4$ affine cover $L_1, \ldots, L_5$ of Section 4.1 (`sbox`);
- the corresponding subsets $\mathcal{S}_1, \ldots, \mathcal{S}_5$ (`points`), also encoded as vanishing constraints (`constraints`);
- the `head_indexes` gatekeeper set, computed by scanning the values $0, \ldots,$ `HEAD_MAX`, with `HEAD_MAX` $= \lfloor p/2^{\omega(m-1)} \rfloor = d_0$ (the leading digit of p). It records the pieces that a have value with the most significant chunk at most d_0, since $x \in \mathbb{F}_p$.

Listing A.1: Affine cover definition.

```
def initPolySbox():
    R.<x,y> = PolynomialRing(F, ["x","y"])
    if BITS == 4:
        sbox = [(y-2*x), (-2*x+3*y-15),
                (y-x+5), (y+2*x-26), (-2*x+5*y+11)]
        points = [[0,1,2,4,5], [3,9,12,15],
                  [10,11,14], [6,7], [8,13]]
        constraints = [(x-0)*(x-1)*(x-2)*(x-4)*(x-5),
                       (x-3)*(x-9)*(x-12)*(x-15),
                       (x-10)*(x-11)*(x-14),
                       (x-6)*(x-7),
                       (x-8)*(x-13)]
        t = len(sbox)
        head_indexes = set()
        idx = 0
        while idx <= HEAD_MAX and len(head_indexes) < t:
            for j in range(len(points)):
                if idx in points[j]:
                    head_indexes.add(j)
                    break
            idx += 1
        head_indexes = sorted(head_indexes)
    return sbox, constraints, head_indexes

HEAD_MAX = P >> (BITS*(M-1))
POLYSBOX, CONSTRAINTS, HEADINDEXES = initPolySbox()
```

The T-box itself is defined directly from Skyscraper's specification:

Listing A.2: Lookup-table T function.

```
def T(z):
    t1 = bitLeftRotation(bitInverse(z), 1)
    t2 = bitLeftRotation(z, 2)
    t3 = bitLeftRotation(z, 3)
    y = z ^^ (t1 & t2 & t3)
    y = bitLeftRotation(y, 1)
    return y
```

A.2 Enumeration Order

Listing A.3 is the enumeration order actually used to produce every result in Chapter 6. As discussed in Section 4.6, only the most-significant chunk of each BAR layer (`head`) is restricted to `HEADINDEXES`, while every other chunk position (`tail`) ranges over all piece indices. Then, the resulting per-layer combinations are combined across the R_B BAR layers via `product`, with no sorting by $\mathbb{P}_{\text{trail}}$ and no reachability filtering. The returned `comb_size` $= (h\ell^{m-1})^{R_B}$ is the size of this enumeration space, where h is the number of head pieces and ℓ^{m-1} the number of tail combinations of each layer with m the number of chunks.

Listing A.3: Enumeration of BAR-layer cases.

```
def genComb():
    lines_num = [i for i in range(len(POLYSBOX))]
    head = HEADINDEXES

    tail = list(product(lines_num, repeat=(STATE_SIZE-1)))
    bar_comb = list(product(head, tail))
    for i in range(len(bar_comb)):
        bar_comb[i] = list(bar_comb[i][:1]) +
              flatten(bar_comb[i][1:], max_level=1)
    comb_size = (((len(lines_num))^(STATE_SIZE-1))
                  *len(head))^BARS
    tot_comb = product(bar_comb, repeat=BARS)
    return tot_comb, comb_size
```

A.3 Attack Loop

Listing A.4 is the hybrid attack loop where

1. is built the enumeration;
2. current choice `c` is substituted into the reduced polynomial system `SkyPoly.cico`;
3. is computed the Gröbner basis of this system;
4. the solution is extracted and verified. If it is valid is returned, otherwise proceeds with next combination `c`.

If the whole enumeration is exhausted without a verified solution the function returns `None`, meaning that no valid solution has been found. However, because the affine cover is an exact case split of the function T, this should never happen and thus it means that something went wrong in the computation.

Listing A.4: Hybrid attack loop.

```
def attack(state = randomCICO()):
    Sky = Skyscraper()
    Sky.digest = Sky.hash(state)

    SkyPoly = SkyscraperPoly()
    tot_comb, comb_size = genComb()

    start_time = time.time()
    i = 0
    for c in tot_comb:
        i += 1
        SkyPoly.cico(Sky.digest, c)

        recovered_input = solve(Sky, SkyPoly)
        if recovered_input is not None:
            time_diff = time.time() - start_time
            return (i, comb_size, time_diff)

    # no valid solution found
    return None
```

The `solve` routine referenced above computes the Gröbner basis of the reduced system and extracts directly the solution, *i.e.*, the CICO input, verifying it exactly before accepting it:

Listing A.5: Reduced system Gröbner basis computation.

```
def solve(Sky, SkyPoly):
    I = SkyPoly.ring.ideal(SkyPoly.equations)
    gb = I.groebner_basis(GB_ALG)

    if gb[0] == 1:
        return None

    if str(gb[0].variables()[0])=='xR0' and gb[0].degree()==1:
        sol = F(-1 * eval(str(gb[0]).replace('xR0', '(0)')))
        if checkSolution(Sky, sol):
            return sol
        else:
            return None

    if (str(gb[0].variables()[0])!='xR0' and
        str(gb[1].variables()[0])=='xR0'):
        var = gb[0].variables()[0]
        factors = factor(gb[0])
        for f in factors:
            sol1 = -1 * eval(
                str(f[0]).replace(str(var), '(0)')
                    .replace('^', '**')
              )
            sol = F(-1 * eval(
                      str(gb[1]).replace('xR0', '(0)')
                        .replace(
                          str(var),'(' + str(sol1) + ')'
                        ).replace('^', '**')
                    ))
            if checkSolution(Sky, sol):
                return sol
        return None
```

Affine Covers

B

This appendix collects the affine covers used for $\omega = 4$ and $\omega = 8$. In both cases, each piece L_i is an affine map $y = \alpha x + \beta$ with coefficients in $\mathbb{F}_p$, read from the rational coefficients given below, and $\mathcal{S}_i \subseteq \{0, \ldots, 2^\omega - 1\}$ is the set of all the inputs x for which $T(x) = \alpha x + \beta$, *i.e.*, the set of points of the lookup table that lie on the line. A family of pieces is a cover of T when the union of the sets $\mathcal{S}_i$ is $\{0, \ldots, 2^\omega - 1\}$.

Cover for $\omega = 4$. The five pieces have pairwise disjoint sets, which therefore partition the domain.

Table B.1: Affine cover for $\omega = 4$.

i	Line L_i	$\lvert\mathcal{S}_i\rvert$	Set $\mathcal{S}_i$
1	$2x$	5	$\{0, 1, 2, 4, 5\}$
2	$2/3x + 5$	4	$\{3, 9, 12, 15\}$
3	$x - 5$	3	$\{10, 11, 14\}$
4	$-2x + 26$	2	$\{6, 7\}$
5	$2/5x - 11/5$	2	$\{8, 13\}$

Cover for $\omega = 8$. The cover is made of $\ell_8 = 42$ pieces and was built with the greedy heuristic of Section 4.2, so it is not guaranteed to be optimal. The sets $\mathcal{S}_i$ below contain all the points of the table that lie on each line, hence they overlap: 35 of the 256 inputs belong to more than one of them. The largest piece is L_1, with $\lvert\mathcal{S}_1\rvert = 34$.

Table B.2: Affine cover for $\omega = 8$.

i	Line L_i	$\lvert\mathcal{S}_i\rvert$	Set $\mathcal{S}_i$
1	$2x$	34	$\{0, 1, 2, 4, 5, 8, 9, 10, 16, 17, 18, 20, 21, 32, 33, 34, 36, 37, 40, 41, 42, 64, 65, 66, 68, 69, 72, 73, 74, 80, 81, 82, 84, 85\}$
2	$2x - 263$	8	$\{133, 149, 165, 197, 213, 229, 245, 253\}$
3	$2x - 259$	8	$\{194, 202, 210, 226, 234, 242, 250, 254\}$
4	$2x - 256$	8	$\{176, 178, 180, 184, 186, 188, 190, 191\}$
5	$2x - 255$	14	$\{128, 130, 132, 136, 138, 144, 146, 148, 160, 162, 164, 168, 170, 255\}$
6	$2x - 251$	12	$\{192, 196, 200, 208, 212, 224, 228, 232, 240, 244, 248, 252\}$
7	$2x - 247$	12	$\{129, 137, 145, 161, 169, 193, 201, 209, 225, 233, 241, 249\}$
8	$2x - 239$	7	$\{131, 147, 163, 195, 211, 227, 243\}$
9	$2x - 128$	7	$\{88, 89, 90, 92, 93, 94, 95\}$
10	$2x - 127$	5	$\{152, 154, 156, 158, 159\}$
11	$2x - 64$	4	$\{44, 45, 46, 47\}$
12	$2x - 62$	2	$\{108, 110\}$
13	$2x - 2$	8	$\{97, 101, 105, 113, 117, 121, 125, 127\}$
14	$2x + 1$	12	$\{48, 49, 50, 52, 53, 56, 57, 58, 60, 61, 62, 63\}$
15	$2x + 2$	12	$\{96, 98, 100, 104, 106, 112, 114, 116, 120, 122, 124, 126\}$
16	$2x + 16$	5	$\{3, 19, 35, 67, 83\}$
17	$2x + 32$	6	$\{6, 7, 38, 39, 70, 71\}$
18	$2x + 64$	8	$\{12, 13, 14, 15, 76, 77, 78, 79\}$
19	$2x + 128$	7	$\{24, 25, 26, 28, 29, 30, 31\}$
20	$-2x + 442$	4	$\{102, 103, 118, 119\}$
21	$2/3x - 257/3$	7	$\{133, 139, 151, 175, 217, 220, 223\}$
22	$2/3x - 5/3$	2	$\{142, 166\}$
23	$2/3x - 4/3$	7	$\{11, 23, 47, 95, 179, 185, 191\}$
24	$2/3x - 1/3$	8	$\{143, 167, 194, 197, 203, 215, 236, 239\}$
25	$2/3x + 85$	11	$\{51, 63, 153, 159, 204, 207, 231, 243, 249, 252, 255\}$
26	$6x - 1275$	13	$\{219, 221, 222, 223, 237, 238, 239, 246, 247, 251, 253, 254, 255\}$
27	$6x - 1019$	10	$\{173, 175, 192, 193, 195, 198, 199, 205, 206, 207\}$
28	$6x - 763$	8	$\{129, 131, 135, 141, 143, 155, 157, 159\}$
29	$-6x + 806$	3	$\{99, 101, 109\}$
30	$-2/5x + 493/5$	3	$\{54, 134, 174\}$
31	$-2/5x + 697/5$	6	$\{51, 141, 161, 171, 216, 221\}$
32	$-2/5x + 884/5$	5	$\{27, 67, 87, 177, 187\}$
33	$6/11x - 757/11$	4	$\{128, 139, 150, 172\}$
34	$11/4x - 289/4$	3	$\{55, 75, 115\}$
35	$-49/37x + 12495/37$	3	$\{107, 181, 218\}$
36	$70/43x$	3	$\{0, 43, 86\}$
37	$38/91x$	3	$\{0, 91, 182\}$
38	$91/37x - 1558/37$	3	$\{22, 59, 96\}$
39	$-7/13x + 2805/13$	3	$\{85, 111, 189\}$
40	$-139/17x + 20973/17$	2	$\{123, 140\}$
41	$59/31x - 8379/31$	2	$\{183, 214\}$
42	$-34/5x + 1797$	2	$\{230, 235\}$

Bibliography

[1] Martin R. Albrecht, Lorenzo Grassi, Christian Rechberger, Arnab Roy, and Tyge Tiessen. "MiMC: Efficient Encryption and Cryptographic Hashing with Minimal Multiplicative Complexity". In: *ASIACRYPT 2016, Part I*. Ed. by Jung Hee Cheon and Tsuyoshi Takagi. Vol. 10031. LNCS. Hanoi, Vietnam: Springer Berlin Heidelberg, Germany, Dec. 2016, pp. 191–219. DOI: 10.1007/978-3-662-53887-6_7.

[2] Martin R. Albrecht, Lorenzo Grassi, Léo Perrin, Sebastian Ramacher, Christian Rechberger, Dragos Rotaru, Arnab Roy, and Markus Schofnegger. "Feistel Structures for MPC, and More". In: *ESORICS 2019, Part II*. Ed. by Kazue Sako, Steve Schneider, and Peter Y. A. Ryan. Vol. 11736. LNCS. Luxembourg: Springer, Cham, Switzerland, Sept. 2019, pp. 151–171. DOI: 10.1007/978-3-030-29962-0_8.

[3] Lorenzo Grassi, Dmitry Khovratovich, Christian Rechberger, Arnab Roy, and Markus Schofnegger. "Poseidon: A New Hash Function for Zero-Knowledge Proof Systems". In: *USENIX Security 2021*. Ed. by Michael Bailey and Rachel Greenstadt. USENIX Association, Aug. 2021, pp. 519–535. URL: https://www.usenix.org/conference/usenixsecurity21/presentation/grassi.

[4] Alan Szepieniec, Tomer Ashur, and Siemen Dhooghe. "Rescue-Prime: a Standard Specification (SoK)". Cryptology ePrint Archive, Report 2020/1143. 2020. URL: https://eprint.iacr.org/2020/1143.

[5] Lorenzo Grassi, Yonglin Hao, Christian Rechberger, Markus Schofnegger, Roman Walch, and Qingju Wang. "Horst Meets Fluid-SPN: Griffin for Zero-Knowledge Applications". In: *CRYPTO 2023, Part III*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14083. LNCS. Santa Barbara, CA, USA: Springer, Cham, Switzerland, Aug. 2023, pp. 573–606. DOI: 10.1007/978-3-031-38548-3_19.

[6] Lorenzo Grassi, Dmitry Khovratovich, Reinhard Lüftenegger, Christian Rechberger, Markus Schofnegger, and Roman Walch. "Reinforced Concrete: A Fast Hash Function for Verifiable Computation". In: *ACM CCS 2022*. Ed. by Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi. Los Angeles, CA, USA: ACM Press, Nov. 2022, pp. 1323–1335. DOI: 10.1145/3548606.3560686.

[7] Lorenzo Grassi, Dmitry Khovratovich, Reinhard Lüftenegger, Christian Rechberger, Markus Schofnegger, and Roman Walch. "Monolith: Circuit-Friendly Hash Functions with New Nonlinear Layers for Fast and Constant-Time Implementations". In: *IACR Trans. Symm. Cryptol.* 2024.3 (2024), pp. 44–83. DOI: 10.46586/tosc.v2024.i3.44-83.

[8] Clémence Bouvier, Lorenzo Grassi, Dmitry Khovratovich, Katharina Koschatko, Christian Rechberger, Fabian Schmid, and Markus Schofnegger. "Skyscraper: Fast Hashing on Big Primes". In: *IACR TCHES* 2025.2 (2025), pp. 743–780. DOI: 10.46586/tches.v2025.i2.743-780.

[9] Lorenzo Grassi, Katharina Koschatko, and Christian Rechberger. "Poseidon and Neptune: Gröbner Basis Cryptanalysis Exploiting Subspace Trails". In: *IACR Trans. Symm. Cryptol.* 2025.2 (2025), pp. 34–86. DOI: 10.46586/tosc.v2025.i2.34-86.

[10] B. Buchberger. "A theoretical basis for the reduction of polynomials to canonical forms". In: *SIGSAM Bull.* 10.3 (Aug. 1976), pp. 19–29. ISSN: 0163-5824. DOI: 10.1145/1088216.1088219.

[11] Jean-Charles Faugére. "A new efficient algorithm for computing Gröbner bases (F4)". In: *Journal of Pure and Applied Algebra* 139.1 (1999), pp. 61–88. ISSN: 0022-4049. DOI: 10.1016/S0022-4049(99)00005-5.

[12] Jean Charles Faugère. "A new efficient algorithm for computing Gröbner bases without reduction to zero (F5)". In: *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*. ISSAC '02. Lille, France: Association for Computing Machinery, 2002, pp. 75–83. ISBN: 1581134843. DOI: 10.1145/780506.780516.

[13] Martin R. Albrecht, Carlos Cid, Lorenzo Grassi, Dmitry Khovratovich, Reinhard Lüftenegger, Christian Rechberger, and Markus Schofnegger. "Algebraic Cryptanalysis of STARK-Friendly Designs: Application to MARVELLous and MiMC". In: *ASIACRYPT 2019, Part III*. Ed. by Steven D. Galbraith and Shiho Moriai. Vol. 11923. LNCS. Kobe, Japan: Springer, Cham, Switzerland, Dec. 2019, pp. 371–397. DOI: 10.1007/978-3-030-34618-8_13.

[14] Gizem Kara and Oğuz Yayla. "Gröbner Basis Attack on STARK-Friendly Symmetric-Key Primitives: JARVIS, MiMC and GMiMCerf". Cryptology ePrint Archive, Report 2021/1232. 2021. URL: https://eprint.iacr.org/2021/1232.

[15] Jan Ferdinand Sauer and Alan Szepieniec. "SoK: Gröbner Basis Algorithms for Arithmetization Oriented Ciphers". Cryptology ePrint Archive, Report 2021/870. 2021. URL: https://eprint.iacr.org/2021/870.

[16] Augustin Bariant, Aurélien Boeuf, Axel Lemoine, Irati Manterola Ayala, Morten Øygarden, Léo Perrin, and Håvard Raddum. "The Algebraic FreeLunch: Efficient Gröbner Basis Attacks Against Arithmetization-Oriented Primitives". In: *CRYPTO 2024, Part IV*. Ed. by Leonid Reyzin and Douglas Stebila. Vol. 14923. LNCS. Santa Barbara, CA, USA: Springer, Cham, Switzerland, Aug. 2024, pp. 139–173. DOI: 10.1007/978-3-031-68385-5_5.

[17] Antoine Bak, Augustin Bariant, Aurélien Boeuf, Pierre Briaud, Morten Øygarden, and Atharva Phanse. "The Algebraic CheapLunch: Extending FreeLunch Attacks on Arithmetization-Oriented Primitives Beyond CICO-1". Cryptology ePrint Archive, Report 2025/2040. 2025. URL: https://eprint.iacr.org/2025/2040.

[18] Augustin Bariant, Aurélien Boeuf, Pierre Briaud, Maël Hostettler, Morten Øygarden, and Håvard Raddum. "Improved Resultant Attack Against Arithmetization-Oriented Primitives". In: *CRYPTO 2025, Part V*. Ed. by Yael Tauman Kalai and Seny F. Kamara. Vol. 16004. LNCS. Santa Barbara, CA, USA: Springer, Cham, Switzerland, Aug. 2025, pp. 335–367. DOI: 10.1007/978-3-032-01901-1_11.

[19] Matthias Johann Steiner. "A Zero-Dimensional Gröbner Basis for Poseidon". Cryptology ePrint Archive, Report 2024/310. 2024. URL: https://eprint.iacr.org/2024/310.

[20] Matthias Johann Steiner. "Zero-Dimensional Gröbner Bases for Rescue-XLIX". Cryptology ePrint Archive, Report 2024/468. 2024. URL: https://eprint.iacr.org/2024/468.

[21] Luca Campa and Arnab Roy. "Gröbner Basis Cryptanalysis of Anemoi". In: *EUROCRYPT 2025, Part I*. Ed. by Serge Fehr and Pierre-Alain Fouque. Vol. 15601. LNCS. Madrid, Spain: Springer, Cham, Switzerland, May 2025, pp. 303–332. DOI: 10.1007/978-3-031-91107-1_11.

[22] Antoine Bak and Léo Perrin. "On the Security of Split-and-Lookup-Based ZK-Friendly Primitives". In: *IACR Trans. Symmetric Cryptol.* 2025.2 (2025), pp. 87–123. DOI: 10.46586/TOSC.V2025.I2.87-123.

[23] Antoine Bak, Guilhem Jazeron, Pierre Galissant, and Léo Perrin. "Attacking Split-and-Lookup-Based Primitives Using Probabilistic Polynomial System Solving Applications to Round-Reduced Monolith and Full-Round Skyscraper". In: *IACR Trans. Symm. Cryptol.* 2025.3 (2025), pp. 337–367. DOI: 10.46586/tosc.v2025.i3.337-367.

[24] Fukang Liu, Katharina Koschatko, Lorenzo Grassi, Hailun Yan, Shiyao Chen, Subhadeep Banik, and Willi Meier. "Opening the Blackbox: Collision Attacks on Round-Reduced Tip5, Tip4, Tip4' and Monolith". In: *IACR Trans. Symm. Cryptol.* 2024.4 (2024), pp. 97–137. DOI: 10.46586/tosc.v2024.i4.97-137.

[25] Antoine Bak. "A practical distinguisher on the full Skyscraper permutation". Cryptology ePrint Archive, Report 2025/102. 2025. URL: https://eprint.iacr.org/2025/102.

[26] Luk Bettale, Jean-Charles Faugère, and Ludovic Perret. "Hybrid approach for solving multivariate systems over finite fields". In: *Journal of Mathematical Cryptology* 3.3 (2009), pp. 177–197. DOI: 10.1515/JMC.2009.009.

[27] Patrizia M. Gianni and Teo Mora. "Algebraic Solution of Systems of Polynomial Equations Using Groebner Bases". In: *Applied Algebra, Algebraic Algorithms and Error Correcting Codes, 5th International Conference, AAECC-5, Menorca, Spain, June 15-19, 1987, Proceedings*. Ed. by Llorenç Huguet i Rotger and Alain Poli. Lecture Notes in Computer Science. Springer, 1987, pp. 247–257. DOI: 10.1007/3-540-51082-6_83.

[28] Alessio Caminata and Elisa Gorla. "Solving Multivariate Polynomial Systems and an Invariant from Commutative Algebra". In: *Arithmetic of Finite Fields - 8th International Workshop, WAIFI*. Ed. by Jean-Claude Bajard and Alev Topuzoglu. Vol. 12542. Lecture Notes in Computer Science. Springer, 2020, pp. 3–36. DOI: 10.1007/978-3-030-68869-1_1.

[29] David G Cantor and Hans Zassenhaus. "A new algorithm for factoring polynomials over finite fields". In: *Mathematics of Computation* 36.154 (1981), pp. 587–592.

[30] Victor Shoup. "Factoring polynomials over finite fields: Asymptotic complexity vs. reality". In: 1993. URL: https://api.semanticscholar.org/CorpusID:15241727.

[31] Anne Canteaut, Tim Beyne, Itai Dinur, Maria Eichlseder, Gregor Leander, Gaëtan Leurent, María Naya Plasencia, Léo Perrin, Yu Sasaki, Yosuke Todo, and Friedrich Wiemer. "Report on the Security of STARK-friendly Hash Functions (Version 2.0)". Technical Report. 2020. URL: https://web.archive.org/web/20260929130951/https://eips.ethereum.org/assets/eip-5988/papers/report_security_stark_friendly_hash.pdf (visited on 09/29/2026).

[32] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. "On the Indifferentiability of the Sponge Construction". In: *EUROCRYPT 2008*. Ed. by Nigel P. Smart. Vol. 4965. LNCS. Istanbul, Turkey: Springer Berlin Heidelberg, Germany, Apr. 2008, pp. 181–197. DOI: 10.1007/978-3-540-78967-3_11.

[33] Nimrod Megiddo and Arie Tamir. "On the complexity of locating linear facilities in the plane". In: *Operations Research Letters* 1.5 (1982), pp. 194–197. DOI: 10.1016/0167-6377(82)90039-6.

[34] Filippo Merlo. "SageMath source code for testing Master thesis affine cover attack against Skyscraper Hash function." In: 2026. URL: https://web.archive.org/web/20260930153708/https://github.com/Crisis82/thesis-code/ (visited on 09/30/2026).