Algebraic Attacks: Theoretical Aspects

Aurélien Bœuf,[1] Léo Perrin[1]

[1]Inria, France

March 13, 2025

In this Presentation

Anemoi	**Griffin**	**ArionHash**
Crypto23	Crypto23	arXiv

In this Presentation

Anemoi
Crypto23

~~**Griffin**~~
~~Crypto23~~

ArionHash
arXiv

Full-round break
of some instances

In this Presentation

Anemoi
Crypto23

~~**Griffin**~~
~~Crypto23~~

Full-round break
of some instances

~~**ArionHash**~~
~~arXiv~~

Full-round break
of some instances

In this Presentation

~~**Anemoi**~~
~~Crypto23~~

Maybe full-round break?

~~**Griffin**~~
~~Crypto23~~

Full-round break
of some instances

~~**ArionHash**~~
~~arXiv~~

Full-round break
of some instances

In this Presentation

~~Anemoi~~
~~Crypto23~~

Maybe full-round break?

~~Griffin~~
~~Crypto23~~

Full-round break
of some instances

~~ArionHash~~
~~arXiv~~

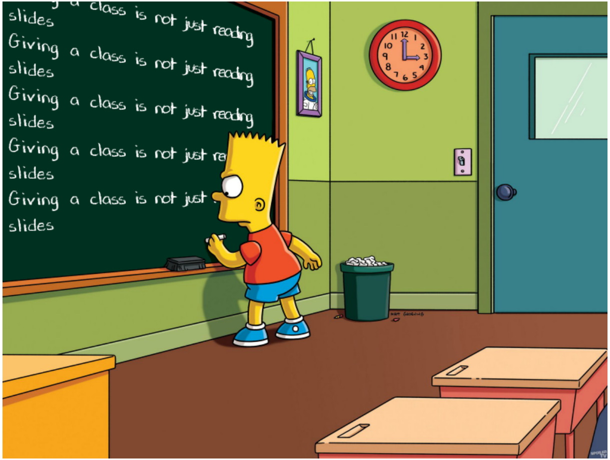
Full-round break
of some instances

How did this happen?

Outline

Plan of this Section

Algebraic Attack? What is that?



Pipeline of a Root-Finding Attack

1. Design an attack

Pipeline of a Root-Finding Attack

1. Design an attack
2. Write a (system of) equation(s)

Pipeline of a Root-Finding Attack

1. Design an attack
2. Write a (system of) equation(s)
3. ???

Pipeline of a Root-Finding Attack

1. Design an attack
2. Write a (system of) equation(s)
3. ???
4. Deduce a preimage/CICO solution/master key...

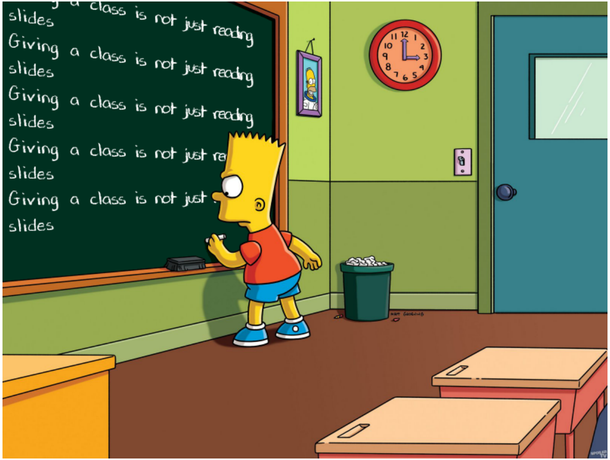
Pipeline of a Root-Finding Attack

1. Design an attack
2. Write a (system of) equation(s)
3. ???
4. Deduce a preimage/CICO solution/master key...

The topic of this class: the “???” part!

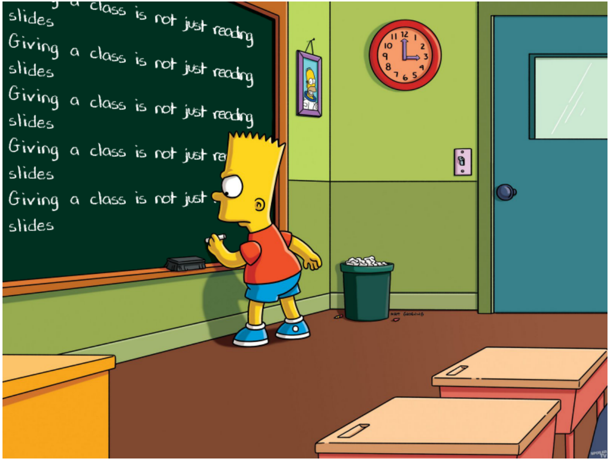
Plan of this Section

Let's Look at Feistel-MiMC



Plan of this Section

Deriving a Multi-Variate System: CICO-2



Plan of this Section

Plan of this Section

Root Finding: Simple Cases

Consider a multivariate polynomial ring $\mathbb{F}[x_1, x_2, \dots, x_N]$.
We want to solve:

$$\begin{cases} p_1(x_1, \dots, x_N) = 0 \\ p_2(x_1, \dots, x_N) = 0 \\ \vdots \\ p_k(x_1, \dots, x_N) = 0 \end{cases}$$

Root Finding: Simple Cases

$$\begin{cases} m_{1,1}x_1 + \cdots + m_{1,N}x_N + a_1 = 0 \\ m_{2,1}x_1 + \cdots + m_{2,N}x_N + a_2 = 0 \\ \vdots \\ m_{k,1}x_1 + \cdots + m_{k,N}x_N + a_k = 0 \end{cases}$$

Polynomials of **degree 1**: Linear system $\Rightarrow$ **Linear algebra**.

Root Finding: Simple Cases

$$\begin{cases} p_1(x_1) = 0 \\ p_2(x_1) = 0 \\ \vdots \\ p_k(x_1) = 0 \end{cases}$$

One variable: Univariate root finding $\Rightarrow$ **Euclidian division** (for Berlekamp-Rabin algorithm).

Root Finding: The Less Simple Case

$$\begin{cases} p_1(x_1, \ldots, x_N) = 0 \\ p_2(x_1, \ldots, x_N) = 0 \\ \vdots \\ p_k(x_1, \ldots, x_N) = 0 \end{cases}$$

Several variables,
high degree!

Root Finding: The Less Simple Case

$$\begin{cases} p_1(x_1, \ldots, x_N) = 0 \\ p_2(x_1, \ldots, x_N) = 0 \\ \qquad \vdots \\ p_k(x_1, \ldots, x_N) = 0 \end{cases}$$

Several variables,
high degree!

General Approach

If $p_i(x_1, \ldots, x_N) = 0$ for all i, then

$$\sum_i q_i(x_1, \ldots, x_N) p_i(x_1, \ldots, x_N) = 0,$$

for any set of polynomials $\{q_i\}_i$.

Root Finding: The Less Simple Case

$$\begin{cases} p_1(x_1,\ldots,x_N) = 0 \\ p_2(x_1,\ldots,x_N) = 0 \\ \vdots \\ p_k(x_1,\ldots,x_N) = 0 \end{cases}$$

Several variables,
high degree!

General Approach

If $p_i(x_1,\ldots,x_N) = 0$ for all i, then

$$\sum_i q_i(x_1,\ldots,x_N)p_i(x_1,\ldots,x_N) \;=\; 0,$$

for any set of polynomials $\{q_i\}_i$.

The set of all such linear combinations is
the ideal generated by the $\{p_i\}_{i=1}^{k}$

We denote it $I(p_0, \ldots, p_{n-1})$.

Root Finding: The Less Simple Case

$$\begin{cases} p_1(x_1, \ldots, x_N) = 0 \\ p_2(x_1, \ldots, x_N) = 0 \\ \vdots \\ p_k(x_1, \ldots, x_N) = 0 \end{cases}$$

Several variables,
high degree!

General Approach

If $p_i(x_1, \ldots, x_N) = 0$ for all i, then

$$\sum_i q_i(x_1, \ldots, x_N) p_i(x_1, \ldots, x_N) = 0,$$

for any set of polynomials $\{q_i\}_i$.

The set of all such linear combinations is the ideal generated by the $\{p_i\}_{i=1}^k$

We denote it $I(p_0, \ldots, p_{n-1})$.

Goal: somehow, find a polynomial $r(x_1) = 0$ in this ideal!

Plan of this Section

Using the Ideal Structure

Over the integer

There is an ideal you all know: $n\mathbb{Z} = \{\ldots, -3n, -2n, -n, 0, n, 2n, 3n, \ldots\}$.

Using the Ideal Structure

Over the integer

There is an ideal you all know: $n\mathbb{Z} = \{\dots, -3n, -2n, -n, 0, n, 2n, 3n, \dots\}$.

Every $x \in \mathbb{Z}$ can be written as $a + b$, where $a \in n\mathbb{Z}$ and $b \in$

Using the Ideal Structure

Over the integer

There is an ideal you all know: $n\mathbb{Z} = \{\dots, -3n, -2n, -n, 0, n, 2n, 3n, \dots\}$.
Every $x \in \mathbb{Z}$ can be written as $a + b$, where $a \in n\mathbb{Z}$ and $b \in \mathbb{Z}/n\mathbb{Z}$.

Using the Ideal Structure

Over the integer

There is an ideal you all know: $n\mathbb{Z} = \{\ldots, -3n, -2n, -n, 0, n, 2n, 3n, \ldots\}$.
Every $x \in \mathbb{Z}$ can be written as $a + b$, where $a \in n\mathbb{Z}$ and $b \in \mathbb{Z}/n\mathbb{Z}$.

We want to simplify our lives and work in

$$\mathbb{F}[x_1, x_2, \ldots, x_N]/I(p_0, \ldots, p_{n-1}) \ .$$

A Problem with Euclidian Division

- Euclidian division on **integers**:

$$a = bq + r\,,\ 0 \leq r < b.$$

Division of 13 by 3:

$$13 = 4 \times 3 + 1.$$

A Problem with Euclidian Division

- Euclidian division on **integers**:

$$a = bq + r\,,\ 0 \leq r < b\,.$$

  Division of 13 by 3:

$$13 = 4 \times 3 + 1.$$

- Euclidian division on **univariate polynomials** ($\mathbb{F}[X]$):

$$A = BQ + R\,,\ \deg(R) < \deg(B).$$

  Division of $X^3 + X + 1$ by X:

$$X^3 + X + 1 = (X^2 + 1)X + 1.$$

The Problem with Multivariate

- Euclidian division on **multivariate polynomials**:

$$A = BQ + R\text{... condition on } R?$$

The Problem with Multivariate

- Euclidian division on **multivariate polynomials**:

$$A = BQ + R\text{... condition on } R?$$

Division of x by $x + y$ in $\mathbb{F}[x, y]$:

$$x = 0 \cdot (x+y) + x$$

or

$$x = 1 \cdot (x+y) - y\ ?$$

The Problem with Multivariate

- Euclidian division on **multivariate polynomials**:

$$A = BQ + R\text{... condition on } R?$$

Division of x by $x + y$ in $\mathbb{F}[x, y]$:

$$x = 0 \cdot (x+y) + x \quad \Leftarrow \quad x < y$$

$$or$$

$$x = 1 \cdot (x+y) - y \quad \Leftarrow \quad y < x$$

Need to define a **monomial ordering**.
$\Longrightarrow$ Division steps determined by **leading monomials (LM)**.

Monomial orderings

In $\mathbb{F}[x, y, z]$:

- **LEXicographical:** Compare degree of highest variable, then second-highest, etc.

$$x <_{\text{lex}} y <_{\text{lex}} z \;, \quad x^{1000} \;\; ? \quad y$$

Monomial orderings

In $\mathbb{F}[x, y, z]$:

- **LEXicographical:** Compare degree of highest variable, then second-highest, etc.

$$x <_{\text{lex}} y <_{\text{lex}} z \;, \quad x^{1000} <_{\text{lex}} y \;, \quad x^6yz \;\; ? \quad y^2z$$

Monomial orderings

In $\mathbb{F}[x, y, z]$:

- **LEXicographical:** Compare degree of highest variable, then second-highest, etc.

$$x <_{\text{lex}} y <_{\text{lex}} z \ , \quad x^{1000} <_{\text{lex}} y \ , \quad x^6 yz <_{\text{lex}} y^2 z \ .$$

Monomial orderings

In $\mathbb{F}[x, y, z]$:

- **LEXicographical:** Compare degree of highest variable, then second-highest, etc.

$$x <_{\text{lex}} y <_{\text{lex}} z \ , \quad x^{1000} <_{\text{lex}} y \ , \quad x^6yz <_{\text{lex}} y^2z \ .$$

- **Graded LEX:** Compare **total degree** first, then switch to lex if equality.

$$x <_{\text{lex}} y <_{\text{lex}} z \ , \quad y \ ? \quad x^2$$

Monomial orderings

In $\mathbb{F}[x, y, z]$:

- **LEXicographical:** Compare degree of highest variable, then second-highest, etc.

$$x <_{\text{lex}} y <_{\text{lex}} z \ , \quad x^{1000} <_{\text{lex}} y \ , \quad x^6 yz <_{\text{lex}} y^2 z \ .$$

- **Graded LEX:** Compare **total degree** first, then switch to lex if equality.

$$x <_{\text{lex}} y <_{\text{lex}} z \ , \quad y <_{\text{glex}} x^2 \ , \quad z^2 \ ? \quad xyz$$

Monomial orderings

In $\mathbb{F}[x, y, z]$:

- **LEXicographical:** Compare degree of highest variable, then second-highest, etc.

$$x <_{\text{lex}} y <_{\text{lex}} z \,,\quad x^{1000} <_{\text{lex}} y \,,\quad x^6 yz <_{\text{lex}} y^2 z \,.$$

- **Graded LEX:** Compare **total degree** first, then switch to lex if equality.

$$x <_{\text{lex}} y <_{\text{lex}} z \,,\quad y <_{\text{glex}} x^2 \,,\quad z^2 <_{\text{glex}} xyz \,,\quad xy <_{\text{glex}} xz <_{\text{glex}} yz \,.$$

Monomial orderings

In $\mathbb{F}[x, y, z]$:

- **LEXicographical:** Compare degree of highest variable, then second-highest, etc.

$$x <_{\text{lex}} y <_{\text{lex}} z \ , \ \ x^{1000} <_{\text{lex}} y \ , \ \ x^6 yz <_{\text{lex}} y^2 z \ .$$

- **Graded LEX:** Compare **total degree** first, then switch to lex if equality.

$$x <_{\text{lex}} y <_{\text{lex}} z \ , \ \ y <_{\text{glex}} x^2 \ , \ \ z^2 <_{\text{glex}} xyz \ , \ \ xy <_{\text{glex}} xz <_{\text{glex}} yz \ .$$

- **Weighted Graded LEX:** Compare the **weighted sum** of degrees, then lex if equality.
  Examples for $x <_{\text{lex}} y <_{\text{lex}} z$ and $\mathbf{wt}(x) = 6, \mathbf{wt}(y) = 1, \mathbf{wt}(z) = 2$:

$$x \quad ? \quad yz^2$$

Monomial orderings

In $\mathbb{F}[x, y, z]$:

- **LEXicographical:** Compare degree of highest variable, then second-highest, etc.

$$x <_{\text{lex}} y <_{\text{lex}} z \ , \quad x^{1000} <_{\text{lex}} y \ , \quad x^6 yz <_{\text{lex}} y^2 z \ .$$

- **Graded LEX:** Compare **total degree** first, then switch to lex if equality.

$$x <_{\text{lex}} y <_{\text{lex}} z \ , \quad y <_{\text{glex}} x^2 \ , \quad z^2 <_{\text{glex}} xyz \ , \quad xy <_{\text{glex}} xz <_{\text{glex}} yz \ .$$

- **Weighted Graded LEX:** Compare the **weighted sum** of degrees, then lex if equality.
  Examples for $x <_{\text{lex}} y <_{\text{lex}} z$ and $\mathbf{wt}(x) = 6, \mathbf{wt}(y) = 1, \mathbf{wt}(z) = 2$:

$$x >_{\text{wglex}} yz^2 \text{ because } \mathbf{wt}(x) = 6 \text{ and } \mathbf{wt}(yz) = \mathbf{wt}(y) + 2\mathbf{wt}(z) = 5 \ .$$

$$x^2 \quad ? \quad z^6$$

Monomial orderings

In $\mathbb{F}[x, y, z]$:

- **LEXicographical:** Compare degree of highest variable, then second-highest, etc.

$$x <_{\text{lex}} y <_{\text{lex}} z \ , \quad x^{1000} <_{\text{lex}} y \ , \quad x^6 yz <_{\text{lex}} y^2 z \ .$$

- **Graded LEX:** Compare **total degree** first, then switch to lex if equality.

$$x <_{\text{lex}} y <_{\text{lex}} z \ , \quad y <_{\text{glex}} x^2 \ , \quad z^2 <_{\text{glex}} xyz \ , \quad xy <_{\text{glex}} xz <_{\text{glex}} yz \ .$$

- **Weighted Graded LEX:** Compare the **weighted sum** of degrees, then lex if equality. Examples for $x <_{\text{lex}} y <_{\text{lex}} z$ and $\mathbf{wt}(x) = 6, \mathbf{wt}(y) = 1, \mathbf{wt}(z) = 2$:

$$x >_{\text{wglex}} yz^2 \text{ because } \mathbf{wt}(x) = 6 \text{ and } \mathbf{wt}(yz) = \mathbf{wt}(y) + 2\mathbf{wt}(z) = 5 \ .$$

$$x^2 < \quad z^6 \text{ because } \mathbf{wt}(x^2) = \mathbf{wt}(z^6) = 12 \text{ and } x^2 < \quad z^6$$

The Problem... Still.

Consider a system $\{p_1, \ldots, p_k\}$.
$\implies$ Division of a polynomial p by $\{p_1, \ldots, p_k\}$ for some ordering: **final remainder can depend on the choice of divisors!**

The Problem... Still.

Consider a system $\{p_1, \ldots, p_k\}$.

$\implies$ Division of a polynomial p by $\{p_1, \ldots, p_k\}$ for some ordering: **final remainder can depend on the choice of divisors!**

Example: in $\mathbb{F}[x, y]$ with **lex** ordering ($x <_{lex} y$), divide y^2 by $\{y^2 - 1, y - x\}$.

$$y^2 \xRightarrow{\text{red. by } y^2 - 1} 1 \xRightarrow{\text{no further red.}} 1$$

$$y^2 \xRightarrow{\text{red. by } y - x} xy \xRightarrow{\text{red. by } y - x} x^2$$

The Problem... Still.

Consider a system $\{p_1, \ldots, p_k\}$.

$\implies$ Division of a polynomial p by $\{p_1, \ldots, p_k\}$ for some ordering: **final remainder can depend on the choice of divisors!**

Example: in $\mathbb{F}[x, y]$ with **lex** ordering ($x <_{lex} y$), divide y^2 by $\{y^2 - 1, y - x\}$.

$$y^2 \xrightarrow{\text{red. by } y^2 - 1} 1 \xrightarrow{\text{no further red.}} 1$$

$$y^2 \xrightarrow{\text{red. by } y - x} xy \xrightarrow{\text{red. by } y - x} x^2$$

The solution: Gröbner Bases.

What is a Gröbner Basis?

Let $G = \{p_1, \ldots, p_k\}$ and $<$ a monomial ordering.

Definition

G is a *Gröbner basis* if and only if the reduction defined by $<$ of any polynomial P does not depend on the order chosen for the reductors.

What is a Gröbner Basis?

Let $G = \{p_1, \ldots, p_k\}$ and $<$ a monomial ordering.

Definition

G is a *Gröbner basis* if and only if the reduction defined by $<$ of any polynomial P does not depend on the order chosen for the reductors.

Useful Proposition

If $LM_<(p_1), \ldots, LM_<(p_k)$ are pairwise ***coprime*** *(e.g. x^2 and y), then G is a Gröbner basis.*

Gröbner Basis - Examples

In $\mathbb{F}[x, y]$:

- $\{y^2 - 1, y - x\}$ is not a Gröbner basis for **lex** order with $x < y$ (previous example).

Gröbner Basis - Examples

In $\mathbb{F}[x, y]$:

- $\{y^2 - 1, y - x\}$ is not a Gröbner basis for **lex** order with $x < y$ (previous example).
- However, it is a Gröbner basis for **lex** order with $x > y$. Proof: $\mathrm{LM}(y^2 - 1) = y^2$ and $\mathrm{LM}(y - x) = x$ are **coprime**.

Gröbner Basis - Examples

In $\mathbb{F}[x, y]$:

- $\{y^2 - 1, y - x\}$ is not a Gröbner basis for **lex** order with $x < y$ (previous example).
- However, it is a Gröbner basis for **lex** order with $x > y$. Proof: $\mathrm{LM}(y^2 - 1) = y^2$ and $\mathrm{LM}(y - x) = x$ are **coprime**.
- $\{y^3 + x, y^3 + x^2\}$ is not a Gröbner basis for any **lex** or **deglex** order.

Gröbner Basis - Examples

In $\mathbb{F}[x, y]$:

- $\{y^2 - 1, y - x\}$ is not a Gröbner basis for **lex** order with $x < y$ (previous example).
- However, it is a Gröbner basis for **lex** order with $x > y$. Proof: $\mathrm{LM}(y^2 - 1) = y^2$ and $\mathrm{LM}(y - x) = x$ are **coprime**.
- $\{y^3 + x, y^3 + x^2\}$ is not a Gröbner basis for any **lex** or **deglex** order.
- However, it is a Gröbner basis for **weighted degree** orders with $\mathbf{wt}(x) = 2$ and $\mathbf{wt}(y) = 1$, as then $\mathrm{LM}(y^3 + x) = y^3$ and $\mathrm{LM}(y^3 + x^2) = x^2$ are **coprime**.

Generic System Solving

$$\begin{cases} p_1(x_1, \ldots, x_N) = 0 \\ \vdots \\ p_{k-1}(x_1, \ldots, x_N) = 0 \\ p_k(x_1, \ldots, x_N) = 0 \end{cases}$$

1. Define system

Generic System Solving

$$\begin{cases} p_1(x_1, \ldots, x_N) = 0 \\ \vdots \\ p_{k-1}(x_1, \ldots, x_N) = 0 \\ p_k(x_1, \ldots, x_N) = 0 \end{cases} \qquad \begin{cases} g_1(x_1, \ldots, x_N) = 0 \\ \vdots \\ g_{\kappa-1}(x_1, \ldots, x_N) = 0 \\ g_\kappa(x_1, \ldots, x_N) = 0 \end{cases}$$

1. Define system
2. Find a GB (F4/F5)

Generic System Solving

$$\begin{cases} p_1(x_1, \ldots, x_N) = 0 \\ \vdots \\ p_{k-1}(x_1, \ldots, x_N) = 0 \\ p_k(x_1, \ldots, x_N) = 0 \end{cases} \qquad \begin{cases} g_1(x_1, \ldots, x_N) = 0 \\ \vdots \\ g_{\kappa-1}(x_1, \ldots, x_N) = 0 \\ g_\kappa(x_1, \ldots, x_N) = 0 \end{cases} \qquad \begin{cases} g_1^*(x_1, \ldots, x_N) = 0 \\ \vdots \\ g_{N-1}^*(x_{N-1}, x_N) = 0 \\ g_N^*(x_N) = 0 \end{cases}$$

1. Define system 2. Find a GB (F4/F5) 3. Change order to **lex** (FGLM)

Generic System Solving

$$\begin{cases} p_1(x_1, \ldots, x_N) = 0 \\ \vdots \\ p_{k-1}(x_1, \ldots, x_N) = 0 \\ p_k(x_1, \ldots, x_N) = 0 \end{cases} \qquad \begin{cases} g_1(x_1, \ldots, x_N) = 0 \\ \vdots \\ g_{\kappa-1}(x_1, \ldots, x_N) = 0 \\ g_\kappa(x_1, \ldots, x_N) = 0 \end{cases} \qquad \begin{cases} g_1^*(x_1, \ldots, x_N) = 0 \\ \vdots \\ g_{N-1}^*(x_{N-1}, x_N) = 0 \\ g_N^*(x_N) = 0 \end{cases}$$

1. Define system
2. Find a GB (F4/F5)
3. Change order to **lex** (FGLM)
4. Find the roots in $\mathbb{F}_q$ of g_N^* with univariate methods, etc.

Generic System Solving

$$\begin{cases} p_1(x_1,\ldots,x_N) = 0 \\ \vdots \\ p_{k-1}(x_1,\ldots,x_N) = 0 \\ p_k(x_1,\ldots,x_N) = 0 \end{cases} \quad \begin{cases} g_1(x_1,\ldots,x_N) = 0 \\ \vdots \\ g_{\kappa-1}(x_1,\ldots,x_N) = 0 \\ g_\kappa(x_1,\ldots,x_N) = 0 \end{cases} \quad \begin{cases} g_1^*(x_1,\ldots,x_N) = 0 \\ \vdots \\ g_{N-1}^*(x_{N-1},x_N) = 0 \\ g_N^*(x_N) = 0 \end{cases}$$

1. Define system
2. Find a GB (F4/F5)
3. Change order to **lex** (FGLM)
4. Find the roots in $\mathbb{F}_q$ of g_N^* with univariate methods, etc.

Remark: Steps 2 and 3 are both computationally costly, but not for the same reasons. For most AOPs, step 2 dominates, **but we can skip it**.

Plan of this Section

Plan of this Section

CICO Problem

CICO Problem of size c (capacity of the sponge) for permutation P:

$$P(*, \ldots, *, \underbrace{0, \ldots, 0}_{c \text{ elements}}) = (*', \ldots, *', \underbrace{0, \ldots, 0}_{c \text{ elements}})$$

CICO Problem

CICO Problem of size c (capacity of the sponge) for permutation P:

$$P(*, \ldots, *, \underbrace{0, \ldots, 0}_{c \text{ elements}}) = (*', \ldots, *', \underbrace{0, \ldots, 0}_{c \text{ elements}})$$

Solving CICO of size c gives collisions to the hash function.

$\Rightarrow$ Multivariate attack: solve CICO faster than brute-force attacks using a model of P.
$\Rightarrow$ We focus on $c = 1$.

$$P(x, *, \ldots, *, 0) = (*', \ldots, *', 0).$$

Block Cipher

Figure: The ever-popular Block Cipher construction.

Quick Overview of Griffin, Arion, Anemoi

Our targets:

Anemoi	**Griffin**	**ArionHash**
Crypto23	Crypto23	arXiv

- Griffin, ArionHash and AnemoiSponge are Arithmetization-Oriented families of hash functions.
- Based on Griffin-π, Arion-π and Anemoi families of permutations (all fixed-key block ciphers).

Quick Overview of Griffin, Arion, Anemoi

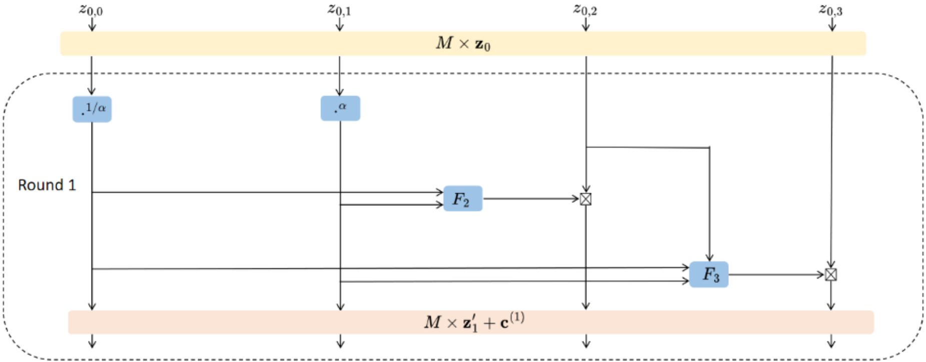
Our targets:

Anemoi	**Griffin**	**ArionHash**
Crypto23	Crypto23	arXiv

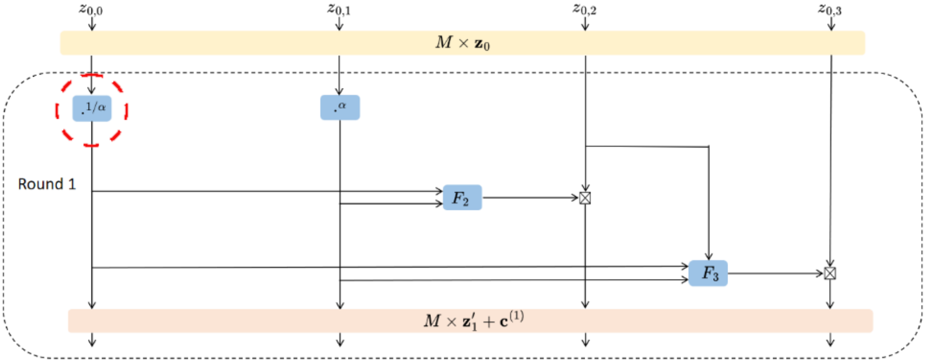
- Griffin, ArionHash and AnemoiSponge are Arithmetization-Oriented families of hash functions.
- Based on Griffin-π, Arion-π and Anemoi families of permutations (all fixed-key block ciphers).
- **Many** instances are defined: variable $\mathbb{F}_p$, number of branches, exponents for monomial permutations...

$\Longrightarrow$ We attack some instance better than others.

Griffin-π - Round Function (4 branches)



Griffin-π - Round Function (4 branches)



$\cdot^{1/\alpha}$ is the only high-degree operation $\implies$ add one variable per $\cdot^{1/\alpha}$.

Griffin-π - Model

- **CICO problem**: $\mathcal{G}_\pi(\cdots||0) = (\cdots||0)$.
  $\implies$ One variable x_0 in the input. One equation for the output (last branch at 0).
- N_{rounds} equations of the form $x_i^\alpha = P_i(x_0, x_1, \ldots x_{i-1})$ $(\cdot^{1/\alpha}$ S-boxes).

Griffin-π - Model

- **CICO problem**: $\mathcal{G}_\pi(\cdots||0) = (\cdots||0)$.
  $\implies$ One variable x_0 in the input. One equation for the output (last branch at 0).
- N_{rounds} equations of the form $x_i^\alpha = P_i(x_0, x_1, \ldots x_{i-1})$ ($\cdot^{1/\alpha}$ S-boxes).

Example ($\alpha = 3$, one round)

$$x_1^3 = ax_0 + b$$
$$x_0^7 + cx_0^4 x_1 + dx_0 x_1^2 + \cdots = 0$$

Generic System Solving

$$\begin{cases} p_1(x_1,\ldots,x_N) = 0 \\ \vdots \\ p_{k-1}(x_1,\ldots,x_N) = 0 \\ p_k(x_1,\ldots,x_N) = 0 \end{cases} \quad \begin{cases} g_1(x_1,\ldots,x_N) = 0 \\ \vdots \\ g_{\kappa-1}(x_1,\ldots,x_N) = 0 \\ g_\kappa(x_1,\ldots,x_N) = 0 \end{cases} \quad \begin{cases} g_1^*(x_1,\ldots,x_N) = 0 \\ \vdots \\ g_{N-1}^*(x_{N-1},x_N) = 0 \\ g_N^*(x_N) = 0 \end{cases}$$

1. Define system
2. Find a GB (F4/F5)
3. Change order to **lex** (FGLM)
4. Find the roots in $\mathbb{F}_q$ of g_N^* with univariate methods, etc.

Designers of Anemoi and Griffin base their security on the hardness of **Step 2**.

Generic System Solving

$$\begin{cases} p_1(x_1,\ldots,x_N) = 0 \\ \vdots \\ p_{k-1}(x_1,\ldots,x_N) = 0 \\ p_k(x_1,\ldots,x_N) = 0 \end{cases}$$

1. Define system

$$\cancel{\begin{cases} g_1(x_1,\ldots,x_N) = 0 \\ \vdots \\ g_{\kappa-1}(x_1,\ldots,x_N) = 0 \\ g_\kappa(x_1,\ldots,x_N) = 0 \end{cases}}$$

2. ~~Find a GB (F4/F5)~~

$$\begin{cases} g_1^*(x_1,\ldots,x_N) = 0 \\ \vdots \\ g_{N-1}^*(x_{N-1},x_N) = 0 \\ g_N^*(x_N) = 0 \end{cases}$$

3. Change order to **lex** (FGLM)

4. Find the roots in $\mathbb{F}_q$ of g_N^* with univariate methods, etc.

Designers of Anemoi and Griffin base their security on the hardness of **Step 2**.

But we can skip it!

Plan of this Section

Griffin-π - Model

- **CICO problem**: $\mathcal{G}_\pi(\cdots||0) = (\cdots||0)$.
  $\implies$ One variable x_0 in the input. One equation for the output (last branch at 0).
- N_{rounds} equations of the form $x_i^\alpha = P_i(x_0, x_1, \ldots x_{i-1})$ $\quad(\cdot^{1/\alpha}$ S-boxes).

Example ($\alpha = 3$, one round)

$$x_1^3 = a x_0 + b$$
$$x_0^7 + c x_0^4 x_1 + d x_0 x_1^2 + \cdots = 0$$

Griffin-π - Model

- **CICO problem**: $\mathcal{G}_\pi(\cdots||0) = (\cdots||0)$.
  $\implies$ One variable x_0 in the input. One equation for the output (last branch at 0).
- N_{rounds} equations of the form $x_i^\alpha = P_i(x_0, x_1, \dots x_{i-1})$ $\quad$ ($\cdot^{1/\alpha}$ S-boxes).

Example ($\alpha = 3$, one round)

$$x_1^3 = ax_0 + b$$
$$x_0^7 + cx_0^4 x_1 + dx_0 x_1^2 + \cdots = 0$$

***Observation**: x_1 has a lower degree than x_0 in the last equation.*

$\implies$ *In **grevlex**, the leading monomials are x_0^7 and x_1^3.*

$\implies$ ***It's a Gröbner basis** ! (coprime leading monomials)*

Griffin-π - Model

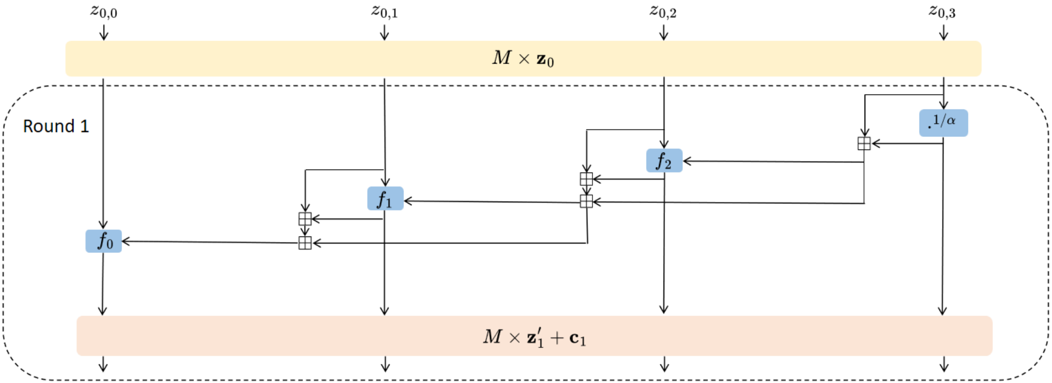
- **CICO problem**: $\mathcal{G}_\pi(\cdots||0) = (\cdots||0)$.
  $\Longrightarrow$ One variable x_0 in the input. One equation for the output (last branch at 0).
- N_{rounds} equations of the form $x_i^\alpha = P_i(x_0, x_1, \ldots x_{i-1})$ $\quad$ ($\cdot^{1/\alpha}$ S-boxes).

Example ($\alpha = 3$, one round)

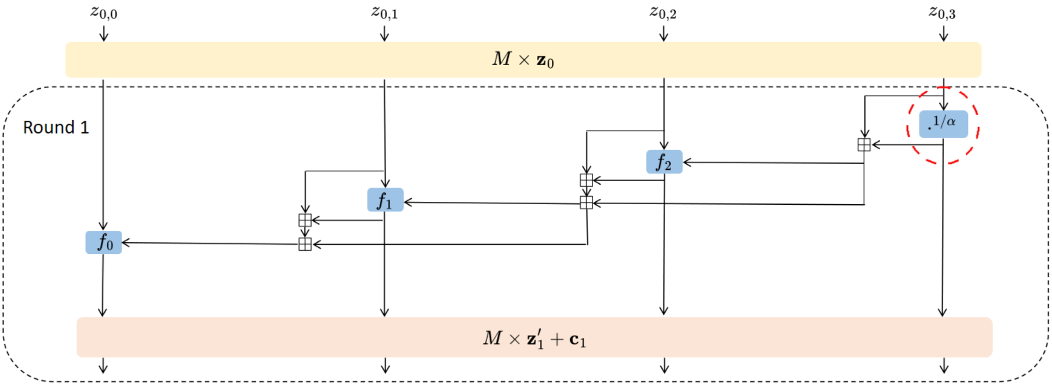
$$x_1^3 = ax_0 + b$$
$$x_0^7 + cx_0^4 x_1 + dx_0 x_1^2 + \cdots = 0$$

Observation: *x_1 has a lower degree than x_0 in the last equation.*
$\Longrightarrow$ *In **grevlex**, the leading monomials are x_0^7 and x_1^3.*
$\Longrightarrow$ ***It's a Gröbner basis** ! (coprime leading monomials)*
*For more rounds, **grevlex** doesn't work. We need **weighted degree** orders, with* $\mathbf{wt}(x_0) = 1$ *and* $\mathbf{wt}(x_i) = 7^{i-1}$.

Arion-π - Round Function (4 branches)



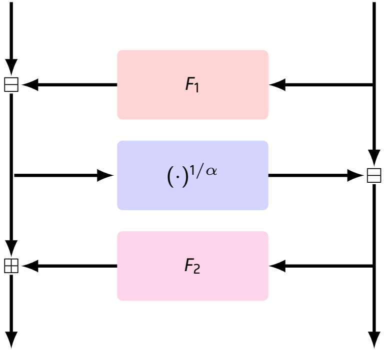
Arion-π - Round Function (4 branches)



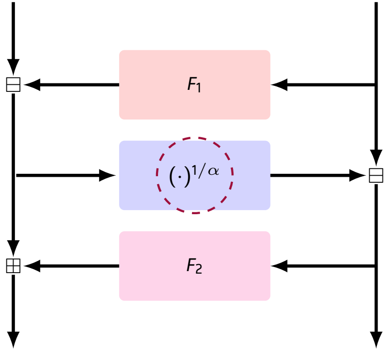
$\cdot^{1/\alpha}$ is the only high-degree operation $\implies$ add one variable per $\cdot^{1/\alpha}$.

Plan of this Section

Anemoi - Nonlinear layer (2 branches)



Anemoi - Nonlinear layer (2 branches)



$(\cdot)^{1/\alpha}$ is the only high-degree operation $\Longrightarrow$ add one variable per $(\cdot)^{1/\alpha}$.

Anemoi - Model

Example ($\alpha = 3$, one round)

$$x_1^3 = a x_0^2 + b x_0 + c$$
$$x_0 x_1 + d x_1^2 + e x_0 + f x_1 + g = 0$$

Anemoi - Model

Example ($\alpha = 3$, one round)

$$x_1^3 = ax_0^2 + bx_0 + c$$
$$x_0x_1 + dx_1^2 + ex_0 + fx_1 + g = 0$$

x_0^2 cancels out: this isn't a Gröbner basis for any order!

Anemoi - Model

Example ($\alpha = 3$, one round)

$$x_1^3 = ax_0^2 + bx_0 + c$$
$$x_0x_1 + dx_1^2 + ex_0 + fx_1 + g = 0$$

x_0^2 cancels out: this isn't a Gröbner basis for any order!

Saving the Freelunch

Solution: multiply last equation by x_1^2 and reduce it by the first equation. We get:

$$p^*(x_0, x_1) = ax_0^3 + bdx_0^2x_1 + \cdots$$

Anemoi - Model

Example ($\alpha = 3$, one round)

$$x_1^3 = a x_0^2 + b x_0 + c$$

$$x_0 x_1 + d x_1^2 + e x_0 + f x_1 + g = 0$$

x_0^2 cancels out: this isn't a Gröbner basis for any order!

Saving the Freelunch

Solution: multiply last equation by x_1^2 and reduce it by the first equation. We get:

$$p^*(x_0, x_1) = a x_0^3 + b d x_0^2 x_1 + \cdots$$

$\Longrightarrow$ The first equation and p^* are a Gröbner basis for some weighted order.

Anemoi - Model

Example ($\alpha = 3$, one round)

$$x_1^3 = ax_0^2 + bx_0 + c$$
$$x_0x_1 + dx_1^2 + ex_0 + fx_1 + g = 0$$

x_0^2 cancels out: this isn't a Gröbner basis for any order!

Saving the Freelunch

Solution: multiply last equation by x_1^2 and reduce it by the first equation. We get:

$$p^*(x_0, x_1) = ax_0^3 + bdx_0^2x_1 + \cdots$$

$\Longrightarrow$ The first equation and p^* are a Gröbner basis for some weighted order.

This adds a few parasitic solutions (corresponding to $x_1 = 0$), but not many.

Plan of this Section

FGLM in a Nutshell

- Given a zero-dimensional ideal I, a Gröbner basis G_1 for I some ordering $<_1$, and an ordering $<_2$, FGLM computes a Gröbner basis G_2 for $<_2$ in $O(n_{var} D_I^3)$.
- D_I is the degree of the ideal, a.k.a. the number of **solutions of the system** in the algebraic closure.

FGLM in a Nutshell

- Given a zero-dimensional ideal I, a Gröbner basis G_1 for I some ordering $<_1$, and an ordering $<_2$, FGLM computes a Gröbner basis G_2 for $<_2$ in $O(n_{var}D_I^3)$.
- D_I is the degree of the ideal, a.k.a. the number of **solutions of the system** in the algebraic closure.
- **This is interesting** because a GB in **lex** order **must have** a univariate polynomial in the smallest variable, which we can solve. (This corresponds to eliminating the other variables.)

FGLM in a Nutshell

- Given a zero-dimensional ideal I, a Gröbner basis G_1 for I some ordering $<_1$, and an ordering $<_2$, FGLM computes a Gröbner basis G_2 for $<_2$ in $O(n_{var}D_I^3)$.
- D_I is the degree of the ideal, a.k.a. the number of **solutions of the system** in the algebraic closure.
- **This is interesting** because a GB in **lex** order **must have** a univariate polynomial in the smallest variable, which we can solve. (This corresponds to eliminating the other variables.)
- Free Gröbner basis, FGLM and symmetric techniques to bypass the first rounds is already enough to break some instances of Griffin and Arion.

Faster Change of Order Strategy

- Idea from a 2022 paper by Jérémy Berthomieu, Vincent Neiger, Mohab Safey El Din.
- Strategy: for the smallest variable x, compute the characteristic polynomial χ of the linear operation $P \mapsto \mathrm{Red}_<(x \cdot P, G)$.
- $\chi(x) = 0$. Generically, this is **exactly** the univariate polynomial in x in the reduced GB of I in **lex** order.
- **Issue:** our systems **do not** verify an important property of the original paper.

Computing the Multiplication Matrix

Step 1: Compute the matrix T of the linear operation in $\mathbb{F}[x_0, x_1, \dots, x_N]$ that maps P to $x_0 \cdot P$.

- Need to reduce monomials of the form $x_0^{k_0+1} x_1^{k_1} \cdots x_N^{k_N}$. **We have no tight complexity estimate for this step**.

Computing the Multiplication Matrix

Step 1: Compute the matrix T of the linear operation in $\mathbb{F}[x_0, x_1, \ldots, x_N]$ that maps P to $x_0 \cdot P$.

- Need to reduce monomials of the form $x_0^{k_0+1} x_1^{k_1} \cdots x_N^{k_N}$. **We have no tight complexity estimate for this step**.
- The matrix is sparse. If leading monomials are $x_0^{d_0}, \ldots, x_N^{d_N}$:

$$T_0 = \left(\begin{array}{cccc|c} 0 & 0 & \cdots & 0 & * \\ \hline 1 & 0 & \cdots & 0 & * \\ 0 & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & 1 & * \end{array}\right)$$

$$\underbrace{\hphantom{0 \quad 0 \quad \cdots \quad 0}}_{(d_0-1)d_1 \cdots d_N} \quad \underbrace{\hphantom{*}}_{d_1 \cdots d_N}$$

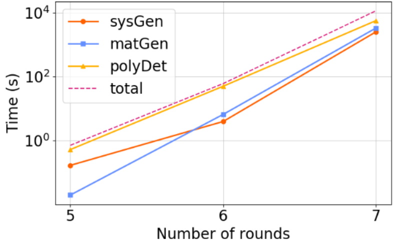
Computing the Characteristic Polynomial

Step 2: Given T, compute $\det(XI - M)$.
$\implies$ T is sparse. With block matrix reasoning, this reduces to computing the determinant of a polynomial matrix of size $D_1 = d_1 \cdots d_N$.
$\implies$ In Griffin and Arion, d_0 is by far the highest degree, so this reduces complexity by a lot.
$\implies$ This can be computed with fast linear algebra, in $\mathcal{O}(d_0 \log(d_0)^2 d_1^\omega \cdots d_N^\omega)$.

Our Full Algorithm

1. `sysGen`: Compute the Freelunch system and the order for a free Gröbner basis.
2. `matGen`: Compute the multiplication matrix T. **Complexity hard to evaluate**.
3. `polyDet`: Compute the characteristic polynomial χ of T.
   $\Longrightarrow$ **Longest step aside from** `matGen`.
4. `uniSol`: Find roots of χ with Berlekamp-Rabin in $\mathcal{O}(D_I \log(D_I) \log(pD_I))$.

Experimental Results



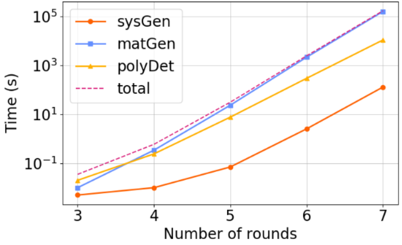
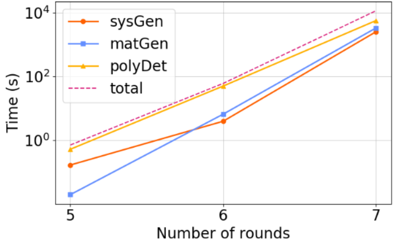


Figure: Complexity of Griffin (7 out of 10 rounds, α=3) Complexity of Anemoi (7 out of 21 rounds, $\alpha = 3$)

Experimental Results



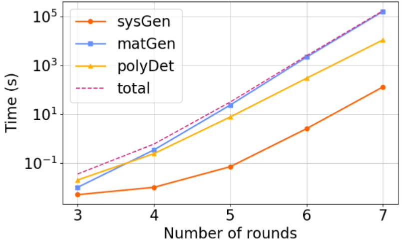


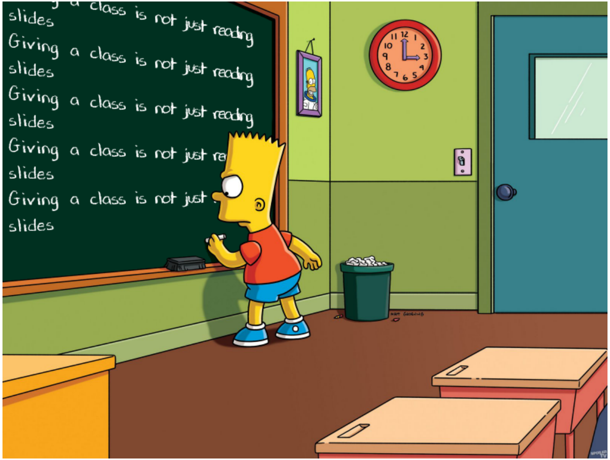
Figure: Complexity of Griffin (7 out of 10 rounds, α=3) | Complexity of Anemoi (7 out of 21 rounds, $\alpha = 3$)

$\Longrightarrow$ For Griffin, `polyDet` upper-bounds the others up to 7 rounds.

$\Longrightarrow$ For Anemoi, `matGen` is the bottleneck.

Plan of this Section

Skipping Rounds



Conclusion

- AO primitives should not base their security on the complexity of finding a Gröbner basis (F4/F5).
- Instead, focus on the growth of D_I with the number of rounds (impacts the complexity of solving algorithms).
- Anemoi, Griffin and Arion need to recompute their numbers of rounds!

Conclusion

- AO primitives should not base their security on the complexity of finding a Gröbner basis (F4/F5).
- Instead, focus on the growth of D_I with the number of rounds (impacts the complexity of solving algorithms).
- Anemoi, Griffin and Arion need to recompute their numbers of rounds!

`MatGen` and `PolyDet` are already getting obsolete!

Improved Resultant Attack against Arithmetization-Oriented Primitives

`https://eprint.iacr.org/2025/259`

Augustin Bariant, Aurélien Boeuf, Pierre Briaud, Maël Hostettler, Morten Øygarden, Håvard Raddum

Conclusion

- AO primitives should not base their security on the complexity of finding a Gröbner basis (F4/F5).
- Instead, focus on the growth of D_I with the number of rounds (impacts the complexity of solving algorithms).
- Anemoi, Griffin and Arion need to recompute their numbers of rounds!

MatGen and PolyDet are already getting obsolete!

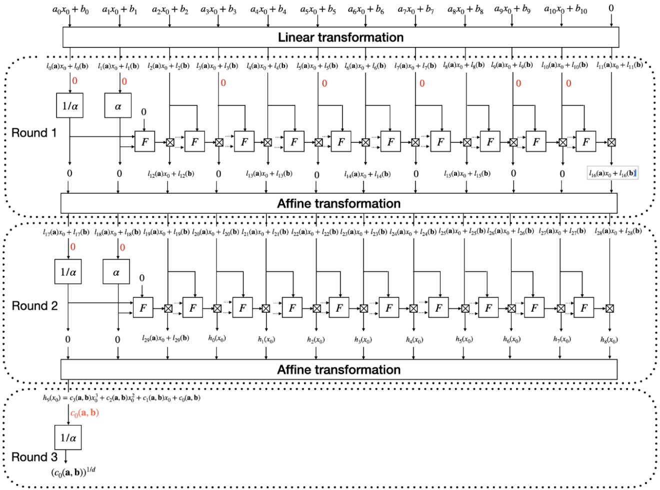
Improved Resultant Attack against Arithmetization-Oriented Primitives

https://eprint.iacr.org/2025/259

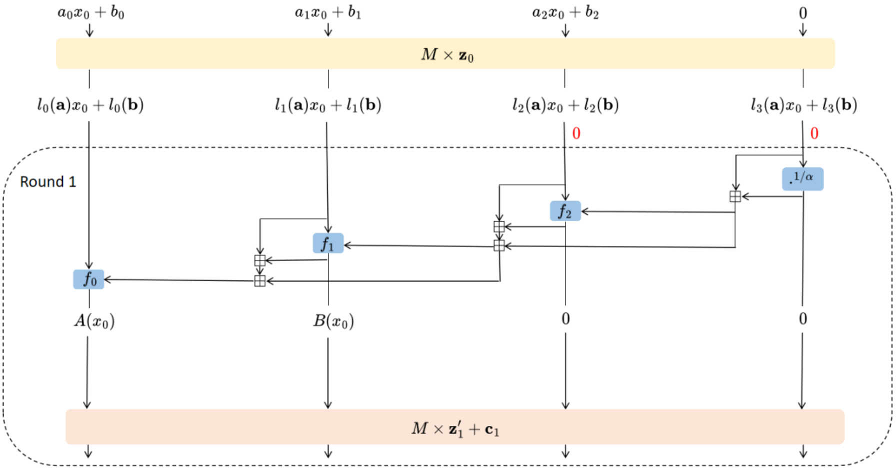
Augustin Bariant, Aurélien Boeuf, Pierre Briaud, Maël Hostettler, Morten Øygarden, Håvard Raddum

Thank you!

Griffin Trick



Arion Trick



$a_0x_0 + b_0$
$a_1x_0 + b_1$
$a_2x_0 + b_2$
0
$M \times \mathbf{z}_0$
$l_0(\mathbf{a})x_0 + l_0(\mathbf{b})$
$l_1(\mathbf{a})x_0 + l_1(\mathbf{b})$
$l_2(\mathbf{a})x_0 + l_2(\mathbf{b})$
$l_3(\mathbf{a})x_0 + l_3(\mathbf{b})$
0
0
Round 1
$.^{1/\alpha}$
f_2
f_1
f_0
$A(x_0)$
$B(x_0)$
0
0
$M \times \mathbf{z}'_1 + \mathbf{c}_1$