Algebraic Attacks against Some Arithmetization-Oriented Primitives

Clémence Bouvier

joint work with Augustin Bariant, Gaëtan Leurent, Léo Perrin

1Sorbonne Université, 2Inria Paris

FSE, March, 2023
AOP: “Appellation d’origine protégée”

Camembert de Normandie
Motivation

A Cryptanalysis Challenge for ZK-friendly Hash Functions!
In November 2021, by the Ethereum Foundation.

| Category | Parameters | Security Level | Bounty  |
|----------|------------|----------------|---------|
| Easy     | $N = 4, m = 3$ | 25             | $2,000  |
| Easy     | $N = 6, m = 2$ | 25             | $4,000  |
| Medium   | $N = 7, m = 2$ | 29             | $6,000  |
| Hard     | $N = 5, m = 3$ | 30             | $12,000 |
| Hard     | $N = 8, m = 2$ | 33             | $26,000 |

(a) Rescue–Prime

| Category | Parameters | Security Level | Bounty  |
|----------|------------|----------------|---------|
| Easy     | $p = 281474976710597$ | 24             | $4,000  |
| Medium   | $p = 72057594037926839$ | 28             | $6,000  |
| Hard     | $p = 18446744073709551557$ | 32             | $12,000 |

(b) Feistel–MiMC

(c) Poseidon
## Motivation

A Cryptanalysis Challenge for ZK-friendly Hash Functions!
In November 2021, by the Ethereum Fundation.

### (a) Rescue–Prime

| Category | Parameters | Security Level | Bounty |
|----------|------------|----------------|--------|
| Easy     | $N = 4, m = 3$ | 25             | $2,000 |
| Easy     | $N = 6, m = 2$ | 25             | $4,000 |
| Medium   | $N = 7, m = 2$ | 29             | $6,000 |
| Hard     | $N = 5, m = 3$ | 30             | $12,000|
| Hard     | $N = 8, m = 2$ | 33             | $26,000|

### (b) Feistel–MiMC

| Category | Parameters | Security Level | Bounty |
|----------|------------|----------------|--------|
| Easy     | $r = 6$    | 6              | $2,000 |
| Easy     | $r = 10$   | 15             | $4,000 |
| Medium   | $r = 14$   | 22             | $6,000 |
| Hard     | $r = 18$   | 28             | $12,000|
| Hard     | $r = 22$   | 34             | $26,000|

### (c) Poseidon

| Category | Parameters | Security Level | Bounty |
|----------|------------|----------------|--------|
| Easy     | $p = 281474976710597$ | 24             | $4,000 |
| Medium   | $p = 72057594037926839$ | 28             | $6,000 |
| Hard     | $p = 18446744073709551557$ | 32             | $12,000|

### (d) Reinforced Concrete
Algebraic Attacks against Some Arithmetization-Oriented Primitives.

1 Preliminaries
   - Arithmetization-Oriented Primitives
   - CICO Problem

2 Solving Systems
   - Univariate Systems
   - Multivariate Systems

3 Trick for SPN
   - Applied to Poseidon
   - Applied to Rescue–Prime

4 Ciminion
Preliminaries

- Arithmetization-Oriented Primitives
- CICO Problem

Solving Systems

- Univariate Systems
- Multivariate Systems

Trick for SPN

- Applied to Poseidon
- Applied to Rescue–Prime

Ciminion
### Comparison with “usual” case

#### A new environment

| “Usual” case |
|---------------|
| **Field size:** | $\mathbb{F}_{2^n}$, with $n \approx 4, 8$ (AES: $n = 8$). |
| **Operations:** | logical gates/CPU instructions |

| Arithmetization-friendly |
|--------------------------|
| **Field size:** | $\mathbb{F}_q$, with $q \in \{2^n, p\}$, $p \approx 2^n$, $n \geq 64$ |
| **Operations:** | large finite-field arithmetic |
**Comparison with “usual” case**

**A new environment**

| "Usual" case | Arithmetization-friendly |
|--------------|--------------------------|
| ★ Field size: | ★ Field size: |
| $\mathbb{F}_{2^n}$, with $n \simeq 4, 8$ (AES: $n = 8$). | $\mathbb{F}_q$, with $q \in \{2^n, p\}$, $p \simeq 2^n$, $n \geq 64$ |
| ★ Operations: | ★ Operations: |
| logical gates/CPU instructions | large finite-field arithmetic |

$\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$, with $p$ given by the order of some elliptic curves

| Examples: | ★ Curve BLS12–381 | ★ Curve BLS12–377 |
|-----------|-------------------|-------------------|
| ★ Curve BLS12–381 | $\log_2 p = 255$ | $\log_2 p = 253$ |
| | $p = 5243587517512619047944774050818596583769055250052763$ | $p = 844461749428370424248824938781546531375899335154063$ |
| | $7822603658699938581184513$ | $827935233455917409239041$ |
Comparison with “usual” case

### A new environment

| “Usual” case | Arithmetization-friendly |
|--------------|--------------------------|
| ★ **Field size:**  
$\mathbb{F}_{2^n}$, with $n \simeq 4, 8$ (AES: $n = 8$).  
★ **Operations:**  
logical gates/CPU instructions | ★ **Field size:**  
$\mathbb{F}_q$, with $q \in \{2^n, p\}$, $p \simeq 2^n$, $n \geq 64$  
★ **Operations:**  
large finite-field arithmetic |

### New properties

| “Usual” case | Arithmetization-friendly |
|--------------|--------------------------|
| $y \leftarrow E(x)$  
★ **Optimized for:**  
implementation in software/hardware | $y \leftarrow E(x)$  
★ **Optimized for:**  
integration within advanced protocols  
$y == E(x)$ |
Comparison with “usual” case

A new environment

| “Usual” case | Arithmetization-friendly |
|--------------|--------------------------|
| ★ Field size: | ★ Field size: |
| $\mathbb{F}_{2^n}$, with $n \simeq 4, 8$ (AES: $n = 8$). | $\mathbb{F}_q$, with $q \in \{2^n, p\}$, $p \simeq 2^n$, $n \geq 64$. |
| ★ Operations: | ★ Operations: |
| logical gates/CPU instructions | large finite-field arithmetic |

Decades of Cryptanalysis

| “Usual” case | Arithmetization-friendly |
|--------------|--------------------------|
| ★ Optimized for: | ★ Optimized for: |
| implementation in software/hardware | integration within advanced protocols |

$y \leftarrow E(x)$ and $y == E(x)$

$\leq 5$ years of Cryptanalysis
CICO Problem

Sponge construction.
CICO Problem

CICO: Constrained Input Constrained Output

**Definition**

Let $F : \mathbb{F}_q^t \rightarrow \mathbb{F}_q^t$ and $u < t$. The CICO problem is:

Finding $X, Y \in \mathbb{F}_q^{t-u}$ s.t. $P(X, 0^u) = (Y, 0^u)$.

When $t = 3$, $u = 1$. 

Sponge construction.
CICO Problem

\[ \mathbb{F}_q \:\downarrow \quad m_0 \quad m_1 \quad m_2 \quad \ldots \quad \uparrow \mathbb{F}_q \]

Absorption

\[ \mathbb{F}_t^q \quad P \quad P \quad P \quad \ldots \quad P \]

Squeezing

### CICO: Constrained Input Constrained Output

**Definition**

Let \( F : \mathbb{F}_q^t \rightarrow \mathbb{F}_q^t \) and \( u < t \). The **CICO** problem is:

Finding \( X, Y \in \mathbb{F}_q^{t-u} \) s.t. \( P(X, 0^u) = (Y, 0^u) \).

\[
\begin{align*}
x_0 & \\
x_1 & \\
0 & \\
\downarrow & \\
\downarrow & \\
\downarrow & \\
Y_0 & \\
Y_1 & \\
0 & \\
\end{align*}
\]

when \( t = 3, \ u = 1 \).

### Ethereum Challenges

solving CICO problem for AO primitives with \( q \sim 2^{64} \) prime
Preliminaries
- Arithmetization-Oriented Primitives
- CICO Problem

Solving Systems
- Univariate Systems
- Multivariate Systems

Trick for SPN
- Applied to Poseidon
- Applied to Rescue–Prime

Ciminion
Find the **roots** of a polynomial $P \in \mathbb{F}_q[X]$, with $\deg P = d$.

**Steps:**

1. Compute $Q = X^q - X \mod P$.
   using a double-and-add algorithm.

2. Compute $R = \gcd(P, Q)$.
   $\text{roots}(P) = \text{roots}(R)$ in $\mathbb{F}_q$

3. Factor $R$.
   $\deg(R) \approx 1 \text{ or } 2$ for random $P$

**Cost (in theory):**

\[
\mathcal{O}(d \log(q) \log(d) \log(\log(d)))
\]

\[
O(d \log^2(d) \log(\log(d)))
\]

negligible.
Univariate Solving

Find the roots of a polynomial \( P \in \mathbb{F}_q[X] \), with \( \deg P = d \).

**Steps:**

1. Compute \( Q = X^q - X \mod P \).
   using a double-and-add algorithm.

2. Compute \( R = \gcd(P, Q) \).
   \( \text{roots}(P) = \text{roots}(R) \) in \( \mathbb{F}_q \)

3. Factor \( R \).
   \( \deg(R) \approx 1 \text{ or } 2 \) for random \( P \)

**Cost (in practice):**

| Degree \( d \) | \( 3^{11} \) | \( 3^{15} \) | \( 3^{18} \) |
|----------------|-------------|-------------|-------------|
| Step 1.        | 14s         | 1,433s      | 47,964s     |
| Step 2.        | 7s          | 903s        | 38,693s     |

for random systems

\[ \mathcal{O}( d \cdot \log(d) \cdot (\log(d) + \log(q)) \cdot \log(\log(d)) ) \]
Multivariate Solving

Compute a **Gröbner Basis (GB)** from polynomial equations in $\mathbb{F}_q[X_1, \ldots X_n]$: 

$$ \begin{cases} 
  P_j, j=1,\ldots,n(X_1, \ldots X_n) = 0, \\
  D_{\text{reg}} \leq 1 + \sum_{i=1}^{n} (d_i - 1), \\
  d \leq \prod_{i=1}^{n} d_i 
\end{cases} $$

**Steps:**

1. **F5** algorithm
   - Compute a grevlex order GB.

2. **FGLM** algorithm
   - Convert it into lex order GB.

3. Find the roots in $\mathbb{F}_q^n$ of the GB polynomials using univariate system resolution.

**Cost (in theory):**

$$ \mathcal{O}\left( nD_{\text{reg}} \times \left( \frac{n + D_{\text{reg}} - 1}{D_{\text{reg}}} \right)^\omega \right), \text{ with } 2 \leq \omega \leq 3 $$

for regular systems

$$ \mathcal{O}(nd^3) \text{ or } \mathcal{O}(nd^\omega) $$

$$ \mathcal{O}(d \log^2(d)) $$
Multivariate Solving

Compute a **Gröbner Basis (GB)** from polynomial equations in \( \mathbb{F}_q[X_1, \ldots X_n] \):

\[
\left\{ P_j, j=1, \ldots , n \right\}(X_1, \ldots X_n) = 0, \quad D_{\text{reg}} \leq 1 + \sum_{i=1}^{n} (d_i - 1), \quad d \leq \prod_{i=1}^{n} d_i
\]

**Steps:**

1. **F5 algorithm**
   Compute a grevlex order GB.

2. **FGLM algorithm**
   Convert it into lex order GB.

3. Find the roots in \( \mathbb{F}_q^n \) of the GB polynomials using univariate system resolution.

**In practice:**

| Degree \( d \) | 1024 | 4608 | 16384 |
|----------------|------|------|-------|
| **F4**         | 2.36s | 92.9s | 3,030s |
| **FGLM**       | 18.96s | 1,011s | 32,069s |

for random systems with 4 equations on 4 variables
**Multivariate Solving**

Compute a **Gröbner Basis (GB)** from polynomial equations in $\mathbb{F}_q[X_1, \ldots X_n]$:

$$\left\{ P_j, j=1, \ldots n(X_1, \ldots X_n) = 0, \quad D_{\text{reg}} \leq 1 + \sum_{i=1}^{n}(d_i - 1), \quad d \leq \prod_{i=1}^{n} d_i \right\}$$

**Steps:**

1. **F5 algorithm**
   - Compute a grevlex order GB.

2. **FGLM algorithm**
   - Convert it into lex order GB.

3. Find the roots in $\mathbb{F}_q^n$ of the GB polynomials using univariate system resolution.

**In practice:**

| Degree $d$ | 1024  | 4608  | 16384 |
|------------|-------|-------|-------|
| F4         | 2.36s | 92.9s | 3,030s |
| FGLM       | 18.96s| 1,011s| 32,069s|

for random systems with 4 equations on 4 variables

**Take Away**

Build univariate $\tilde{O}(d)$ instead of multivariate $\tilde{O}(d^3)$ systems when possible!
1 Preliminaries
   - Arithmetization-Oriented Primitives
   - CICO Problem

2 Solving Systems
   - Univariate Systems
   - Multivariate Systems

3 Trick for SPN
   - Applied to Poseidon
   - Applied to Rescue–Prime

4 Ciminion
Let $P = P_0 \circ P_1$ be a permutation of $\mathbb{F}_p^3$ and suppose

$$\exists \ V, \ G \in \mathbb{F}_p^3, \quad \text{s.t.} \ \forall \ X \in \mathbb{F}_p, \quad P_0^{-1}(XV + G) = (\ast, \ast, 0).$$

**Approach used against POSEIDON and Rescue–Prime**
L. Grassi, D. Khovratovich, C. Rechberger, A. Roy
and M. Schofnegger, USENIX 2021

★ SPN construction:
  ★ S-Box layer: \( x \mapsto x^\alpha \), \( \alpha = 3 \)
  ★ Linear layer: MDS
  ★ Round constants addition: AddC

★ Number of rounds (for challenges):

\[
R = 2 \times Rf + RP
= 8 + \text{(from 3 to 24)}.
\]
\begin{equation*}
\begin{aligned}
V &= (A^3, B^3, 0), \\
G &= (0, 0, g),
\end{aligned}
\end{equation*}

with
\begin{equation*}
\begin{aligned}
B &= -\frac{\alpha_{0,2}}{\alpha_{1,2}} A \\
g &= \left(\frac{1}{\alpha_{2,2}} (\alpha_{0,2} c_0^1 + \alpha_{1,2} c_1^1) + c_2^1 + (c_0^2)^3\right)^3.
\end{aligned}
\end{equation*}

\begin{tabular}{|c|c|c|c|c|}
\hline
$R$ & Designers claims & Ethereum estimations & $d$ & complexity \\
\hline
8 + 3 & 2^{17} & 2^{45} & 3^9 & 2^{26} \\
8 + 8 & 2^{25} & 2^{53} & 3^{14} & 2^{35} \\
8 + 13 & 2^{33} & 2^{61} & 3^{19} & 2^{44} \\
8 + 19 & 2^{42} & 2^{69} & 3^{25} & 2^{54} \\
8 + 24 & 2^{50} & 2^{77} & 3^{30} & 2^{62} \\
\hline
\end{tabular}

**Complexity of our attack against Poseidon.**
Rescue–Prime

A. Aly, T. Ashur, E. Ben-Sasson, S. Dhooghe and A. Szepieniec, *ToSC 2020*

- SPN construction:
  - S-Box layer: $x \mapsto x^\alpha$ and $x \mapsto x^{1/\alpha}$, ($\alpha = 3$)
  - Linear layer: MDS
  - Round constants addition: AddC

- Number of rounds (for challenges):
  \[ R = \text{from } 4 \text{ to } 8 \]
  \[ (2 \text{ S-boxes per round}). \]
Rescue–Prime

A. Aly, T. Ashur, E. Ben-Sasson, S. Dhooghe and A. Szepieniec, ToSC 2020

★ SPN construction:
★ S-Box layer: \( x \mapsto x^\alpha \) and \( x \mapsto x^{1/\alpha} \), \( (\alpha = 3) \)
★ Linear layer: MDS
★ Round constants addition: AddC

★ Number of rounds (for challenges):

\[ R = \text{from 4 to 8} \]

(2 S-boxes per round).

Example of parameters

\[
\begin{align*}
 p &= 18446744073709551557 \\
 &\approx 2^{64} \\
 \alpha &= 3 \\
 \alpha^{-1} &= 12297829382473034371
\end{align*}
\]
A. Aly, T. Ashur, E. Ben-Sasson, S. Dhooghe and A. Szepieniec, *ToSC 2020*

- **SPN construction:**
  - S-Box layer: \( x \mapsto x^\alpha \) and \( x \mapsto x^{1/\alpha} \), \( (\alpha = 3) \)
  - Linear layer: MDS
  - Round constants addition: AddC

- Number of rounds (for challenges):
  - \( R = \) from 4 to 8
  - (2 S-boxes per round).
Rescue–Prime

\[
V = (A^3, B^3, 0), \quad G = (0, 0, g),
\]

with

\[
\begin{align*}
B &= -\frac{\alpha_{0,2}}{\alpha_{1,2}} A \\
g &= \left(\frac{1}{\alpha_{2,2}} (\alpha_{0,2} c_0^0 + \alpha_{1,2} c_1^0) + c_2^0\right)^{1/3}.
\end{align*}
\]

| R | m | Designers claims | Ethereum estimations | d | complexity |
|---|---|------------------|----------------------|---|------------|
| 4 | 3 | $2^{36}$         | $2^{37.5}$           | $3^9$ | $2^{43}$  |
| 6 | 2 | $2^{40}$         | $2^{37.5}$           | $3^{11}$ | $2^{53}$  |
| 7 | 2 | $2^{48}$         | $2^{43.5}$           | $3^{13}$ | $2^{62}$  |
| 5 | 3 | $2^{48}$         | $2^{45}$             | $3^{12}$ | $2^{57}$  |
| 8 | 2 | $2^{56}$         | $2^{49.5}$           | $3^{15}$ | $2^{72}$  |

*Complexity of our attack against Rescue.*
Preliminaries
- Arithmetization-Oriented Primitives
- CICO Problem

Solving Systems
- Univariate Systems
- Multivariate Systems

Trick for SPN
- Applied to Poseidon
- Applied to Rescue–Prime

Ciminion
Construction: **Toffoli gates**

\[(a, b, c) \mapsto (a, b, c + ab)\]

**Round function.**

**Overview of Ciminion in \( \mathbb{F}_p \).**
**Attack on Ciminion**

* Designers’ system:
  * 6 equations . . .
  * over 6 variables . . .
  * of degrees
    \[ \{2^{R-1}, 2^R, 2^{R+1}, 2^{R+1}, 2^{R+2}\} \]

\[ \Rightarrow \]

* Our system
  * 4 equations . . .
  * over 4 variables . . .
  * of degrees
    \[ \{2^{R-1}, 2^R, 3 \cdot 2^{R-1}, 3 \cdot 2^{R-1}\} \]

Attack in roughly \(2^{112}\).
Attack on Ciminion

★ Designers’ system:
★ 6 equations . . .
★ over 6 variables . . .
★ of degrees
{2^{R-1}, 2^R, 2^{R+1}, 2^{R+1}, 2^{R+2}}

★ Our system
★ 4 equations . . .
★ over 4 variables . . .
★ of degrees
{2^{R-1}, 2^R, 3 \cdot 2^{R-1}, 3 \cdot 2^{R-1}}

Attack in roughly

\[2^{112.4}\]
Conclusions

Some suggestions for designers:

★ consider as many variants of encoding as possible
★ build univariate instead of multivariate systems when possible
★ start (and end) with a linear layer
★ 2 rounds can be skipped with the trick
Some suggestions for designers:

- consider as many variants of encoding as possible
- build univariate instead of multivariate systems when possible
- start (and end) with a linear layer
- 2 rounds can be skipped with the trick
Conclusions

Some suggestions for designers:

- consider as many variants of encoding as possible
- build univariate instead of multivariate systems when possible
- start (and end) with a linear layer
- 2 rounds can be skipped with the trick

Thanks for your attention
Trick for SPN

\[ M = S_1^{-1}(X) - c_0^1 \]
\[ S_2^{-1}(X) - c_0^1 \]
\[ S_2^{-1}(Y) - c_1^1 \]
\[ S_2^{-1}(Z) - c_2^1 \]

\[ S_1(c_0^0) \]
\[ S_1(c_1^0) \]
\[ S_1(c_2^0) \]

Clémence Bouvier
Algebraic Attacks against Some AOP
Univariate systems: **POSEIDON, Feistel–MiMC**

**Time**
- Feistel-MiMC $X^p \text{ mod } P$
- Feistel-MiMC GCD
- **POSEIDON** $X^p \text{ mod } P$
- **POSEIDON** GCD
- Random $X^p \text{ mod } P$
- Random GCD

- $5.31 \cdot 10^{-5} \cdot d^{1.04}$
- $9.96 \cdot 10^{-6} \cdot d^{1.11}$

**Memory**
- Feistel-MiMC
- **POSEIDON**
- Random

- $7.06 \cdot 10^{-4} \cdot d^{0.98}$
Multivariate systems: Rescue–Prime

![Graph showing time and memory consumption for different methods.]

- **Time**
  - Rescue F4: $3.88 \times 10^{-8} \cdot 2^{4.19n}$
  - Random F4: $3.57 \times 10^{-9} \cdot 2^{5.19n}$
  - Rescue FGLM: $7.48 \times 10^{-9} \cdot 2^{4.99n}$
  - Random FGLM: $9.4 \times 10^{-8} \cdot 2^{4.49n}$

- **Memory**
  - Rescue: $4.08 \times 10^{-4} \cdot 2^{4.01n}$
  - Random: $6.22 \times 10^{-4} \cdot 2^{2.95n}$
Multivariate systems: CIMINION

### Time
- Ciminion F4: $7.12 \cdot 10^{-8} \cdot 2^{8.9r}$
- Random F4: $8.59 \cdot 10^{-8} \cdot 2^{9.99r}$
- Ciminion FGLM: $9 \cdot 10^{-8} \cdot 2^{10.52r}$
- Random FGLM: $3.66 \cdot 10^{-7} \cdot 2^{10.39r}$

### Memory
- Ciminion: $4.17 \cdot 10^{-4} \cdot 2^{4.57r}$
- Random Ciminion: $3.61 \cdot 10^{-4} \cdot 2^{4.87r}$