SCIENCE
PASSION
TECHNOLOGY

RC_p:
Fast Arithmetization-Friendly Hashing

Lorenzo Grassi, Dmitry Khovratovich, Reinhard Lüftenegger,
Christian Rechberger, Markus Schofnegger, **Roman Walch**

23.04.2023

Domain Specific Symmetric Primitives

- Modern cryptographic protocols
  - ZKP: Hash functions in Computational Integrity Proof Systems
  - MPC: Multiple parties jointly compute a function on private input
  - HE: Compute on encrypted data
- Symmetric Primitives are useful in these protocols
- ... but have different design criteria:
  - Prime fields
  - Minimizing multiplicative complexity/depth

$\Rightarrow$ Many new primitives designed

Domain Specific Symmetric Primitives

- Modern cryptographic protocols
  - ZKP: Hash functions in Computational Integrity Proof Systems
  - MPC: Multiple parties jointly compute a function on private input
  - HE: Compute on encrypted data
- Symmetric Primitives are useful in these protocols
- ... but have different design criteria:
  - Prime fields
  - Minimizing multiplicative complexity/depth

$\Rightarrow$ Many new primitives designed

Computational Integrity Proof Systems

- Prove that something has been computed correctly
  - Program, hash function, Merkle-tree
  - Potentially with zero-knowledge
- Many use cases involve hash functions
- Arithmetization
  - Convert program to proof system representation
  - Traditional hash functions often have inefficient representation

$\Rightarrow$ New hash functions:

  - POSEIDON, *Rescue*, GRIFFIN, `Reinforced Concrete`, ...

Design Criteria

- Depends on proof system
  - Low number of multiplication (e.g., R1CS, Plonk)
  - Low-degree representation and low-depth (e.g, AIR)
  - Low number of additions (e.g., original Plonk)
- Recently:
  - Support for lookup tables
- Use cases:
  - Plain performance often bottleneck!

Symmetric Function Concepts in the Past

Type 1

"low degree only"

- Low-degree

$$y = x^d$$

- Fast in Plain
- Many rounds
- Often more constraints
- POSEIDON, Poseidon2, NEPTUNE, GMiMC

Type 2

"non-procedural", "fluid"

- Low-degree equivalence

$$y = x^{1/d} \Rightarrow x = y^d$$

- Slow in Plain
- Fewer rounds
- Fewer constraints
- *Rescue*, GRIFFIN, ANEMOI

Type 3

"lookups"

- Lookup tables

$$y = T[x]$$

- Very fast in Plain
- Even fewer rounds
- Constraints depend on proof system
- `Reinforced Concrete`, Tip5

Symmetric Function Concepts in the Past

Type 1

"low degree only"

- Low-degree

$$y = x^d$$

- Fast in Plain
- Many rounds
- Often more constraints
- POSEIDON, Poseidon2, NEPTUNE, GMiMC

Type 2

"non-procedural", "fluid"

- Low-degree equivalence

$$y = x^{1/d} \Rightarrow x = y^d$$

- Slow in Plain
- Fewer rounds
- Fewer constraints
- *Rescue*, GRIFFIN, ANEMOI

Type 3

"lookups"

- Lookup tables

$$y = T[x]$$

- Very fast in Plain
- Even fewer rounds
- Constraints depend on proof system
- `Reinforced Concrete`, Tip5

Symmetric Function Concepts in the Past

Type 1

"low degree only"

- Low-degree

$$y = x^d$$

- Fast in Plain
- Many rounds
- Often more constraints
- POSEIDON, Poseidon2, NEPTUNE, GMiMC

Type 2

"non-procedural", "fluid"

- Low-degree equivalence

$$y = x^{1/d} \Rightarrow x = y^d$$

- Slow in Plain
- Fewer rounds
- Fewer constraints
- *Rescue*, GRIFFIN, ANEMOI

Type 3

"lookups"

- Lookup tables

$$y = T[x]$$

- Very fast in Plain
- Even fewer rounds
- Constraints depend on proof system
- `Reinforced Concrete`, Tip5

Goal

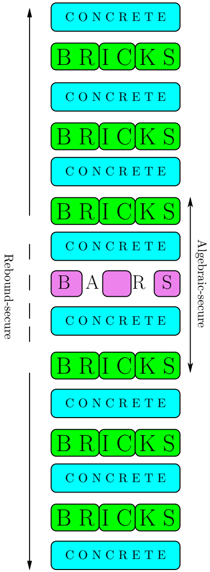
- New Hash function:
  - Efficient plain performance
    - Implementable without lookup tables to resist side-channel attacks
  - Efficient proof system representation
- Focus on FRI-based proof systems
  - Prime-field with fast modular reductions!
    - Particular: $p = 2^{64} - 2^{32} + 1$
  - Allows lookup arguments for less constraints

$\Rightarrow$ RC_p (with $p = 2^{64} - 2^{32} + 1$)

RC_p

Reinforced Concrete

- First arithmetization friendly hash function optimized for lookup tables
- 3 types of layer:
  - Concrete: Linear mixing
  - Bricks: Arithmetic non-linear layer
  - Bars: Decomposition and lookup table
    - Lookup represents repeated $(x + a)^d$
- Security:
  - One Bars layer for algebraic security
  - 6 Concrete/ Bricks for statistical security



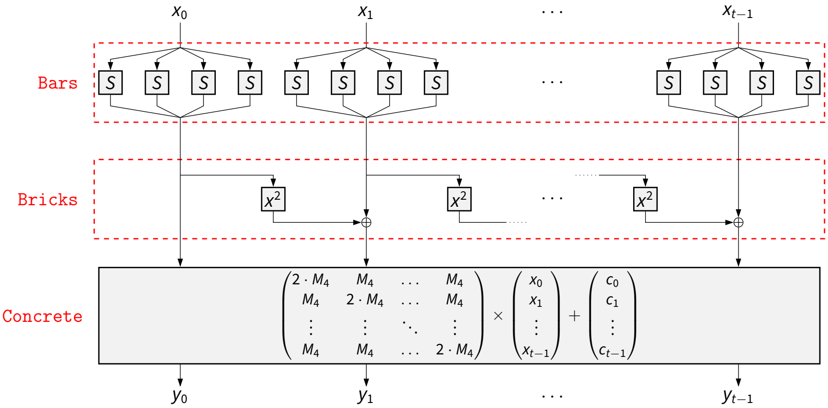
Reinforced Concrete (cont.)

- Faster than any previously published arithmetization oriented hash function
  - When using lookup tables
- But still significantly slower than, e.g., SHA-3
- Problems:
  - Fixed statesize $t = 3$

    $\Rightarrow$ large prime fields ($\log_2(p) = 256$)
  - Decomposition is slow and difficult to generalize
  - Arithmetic function in lookup table
    - Slow without lookup table
    - Only efficient in proof systems with lookup tables

The RC_p Permutation

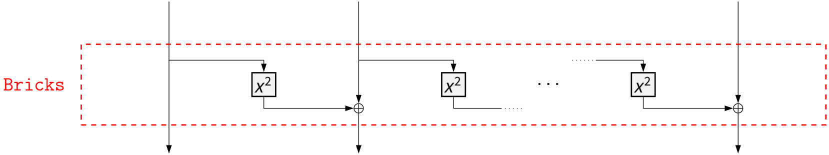
Let statesize $t \geq 8, t = 4 \cdot t'$, one round is given as:



Bricks

- Arithmetic non-linear layer constructed from a quadratic Feistel

$$\texttt{Bricks}(x_0, x_1, \ldots, x_{t-1}) := (x_0, x_1 + x_0^2, x_2 + x_1^2, \ldots, x_{t-1} + x_{t-2}^2).$$



- Cheap in plain
- Cheap in proof systems
  - Small number of multiplication, low-degree polynomials
- Good statistical properties (x^2 has $DP_{max} = 1/p$)

Concrete

- Affine layer $M \cdot x + c^{(i)}$
- Matrix used in Griffin [GHR+22]

$$M = \mathrm{circ}(2 \cdot M_4, M_4, \ldots, M_4) \in \mathbb{F}_p^{t \times t},$$

$$= \begin{bmatrix} 2 \cdot M_4 & M_4 & \ldots & M_4 \\ M_4 & 2 \cdot M_4 & \ldots & M_4 \\ \vdots & & \ddots & \vdots \\ M_4 & M_4 & \ldots & 2 \cdot M_4 \end{bmatrix}$$

…where M_4 is a 4×4 MDS matrix

Concrete (cont.)

- Matrix very cheap in plain
  - M_4 computable by few additions only
  - Also true for full matrix M
- Good statistical properties:
  - Branch number is $t/4+4$

$\Rightarrow$ Together with Bricks provides statistical security

$$M_4 = \begin{pmatrix} 5 & 7 & 1 & 3 \\ 4 & 6 & 1 & 1 \\ 1 & 3 & 5 & 7 \\ 1 & 1 & 4 & 6 \end{pmatrix}$$

Bars

- Binary non-linear layer
- Decompose → S-box → Compose
- Decomposition / Composition

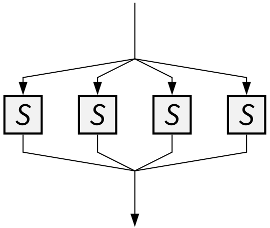
$$x \Leftrightarrow 2^{48}x_3 + 2^{32}x_2 + 2^{16}x_1 + x_0$$

…i.e., split into 16-bit words

- χ-like S-box: $y = S(x)$:

$$S(x) = x \oplus \left((\overline{x} \lll 1) \odot (x \lll 2) \odot (x \lll 3)\right),$$

⇒ Provides algebraic security

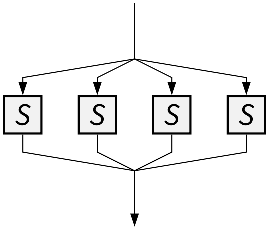


Bars

- Binary non-linear layer
- Decompose $\rightarrow$ S-box $\rightarrow$ Compose
- Decomposition / Composition

$$x \Leftrightarrow 2^{48}x_3 + 2^{32}x_2 + 2^{16}x_1 + x_0$$

…i.e., split into 16-bit words



- χ-like S-box: $y = S(x)$:

$$S(x) = x \oplus ((\overline{x} \lll 1) \odot (x \lll 2) \odot (x \lll 3)),$$

$\Rightarrow$ Provides algebraic security

Bars

- Binary non-linear layer
- Decompose $\rightarrow$ S-box $\rightarrow$ Compose
- Decomposition / Composition

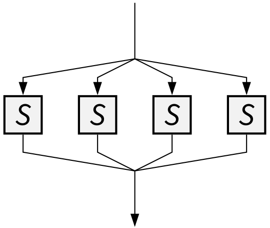
$$x \Leftrightarrow 2^{48}x_3 + 2^{32}x_2 + 2^{16}x_1 + x_0$$

…i.e., split into 16-bit words

- χ-like S-box: $y = S(x)$:

$$S(x) = x \oplus \left((\bar{x} \lll 1) \odot (x \lll 2) \odot (x \lll 3)\right),$$

$\Rightarrow$ Provides algebraic security



$\texttt{Bars}$ (cont.)

- Binary S-box in $\mathbb{F}_p$ hash function
  - Cheap in proof system due to lookup table
  - Cheap in plain due to fast vectorized implementation
  - Provides good algebraic properties
- Well-defined over $\mathbb{F}_p$?
- $p = 2^{64} - 2^{32} + 1$: $\quad p - 1 = 0xFFFF\ FFFF\ 0000\ 0000$
- If $S(0xFFFF) = 0xFFFF$ and $S(0x0000) = 0x0000$:
  - $\texttt{Bars}(p-1) = p - 1$
  - $\texttt{Bars}(x) < p - 1 \qquad \forall x \in \mathbb{F}_p < p - 1$
  - …since nothing except $0xFFFF$ can map to $0xFFFF$

$\texttt{Bars}$ (cont.)

- Binary S-box in $\mathbb{F}_p$ hash function
  - Cheap in proof system due to lookup table
  - Cheap in plain due to fast vectorized implementation
  - Provides good algebraic properties
- Well-defined over $\mathbb{F}_p$?
- $p = 2^{64} - 2^{32} + 1$: $\quad p - 1 = 0xFFFF\ FFFF\ 0000\ 0000$
- If $S(0xFFFF) = 0xFFFF$ and $S(0x0000) = 0x0000$:
  - $\texttt{Bars}(p-1) = p - 1$
  - $\texttt{Bars}(x) < p - 1 \qquad \forall x \in \mathbb{F}_p < p - 1$
  - …since nothing except $0xFFFF$ can map to $0xFFFF$

Security Analysis

(Work in progress)

Algebraic Properties of Bars

- Ideally: Bars represented by dense and high-degree polynomials
- Experiments on smaller, similar primes with $p = 2^n - 2^m + 1$:
  - Bars provides maximum degree ($\approx 2^n$)
  - Density of polynomials $> 99\%$

  $\Rightarrow$ 2 Bars for dense polynomials with degree 2^{128} with $p \approx 2^{64}$

  $\Rightarrow$ 4 Bars required for Meet-in-the-middle attacks
- Lower bounds proven in paper
  - Bars has degree $\geq 2^{57}$ over $\mathbb{F}_p$
  - Bars^{-1} has degree $\geq 2^{47}$ over $\mathbb{F}_p$

  $\Rightarrow$ 6 Bars more than enough

Algebraic Properties of Bars

- Ideally: Bars represented by dense and high-degree polynomials
- Experiments on smaller, similar primes with $p = 2^n - 2^m + 1$:
  - Bars provides maximum degree ($\approx 2^n$)
  - Density of polynomials $> 99\%$

  $\Rightarrow$ 2 Bars for dense polynomials with degree 2^{128} with $p \approx 2^{64}$

  $\Rightarrow$ 4 Bars required for Meet-in-the-middle attacks
- Lower bounds proven in paper
  - Bars has degree $\geq 2^{57}$ over $\mathbb{F}_p$
  - $\texttt{Bars}^{-1}$ has degree $\geq 2^{47}$ over $\mathbb{F}_p$

  $\Rightarrow$ 6 Bars more than enough

Statistical Attacks

- We just consider Concrete and Bricks
- Differential attacks:
  - Each active $x \mapsto x^2$ map has $\text{DP}_{\max} = 1/p$
    - Main issue: $x_0 \mapsto x_0$ in Bricks
  - We show that two consecutive rounds have $\text{DP} \leq p^{-t/8-1/2}$

  $\Rightarrow$ 6 rounds have $\text{DP} \leq 2^{-256}$ (for $t \geq 8$)
- Other attacks:
  - Rebound attack, truncated differential attacks, ...
  - Work in progress...

Statistical Attacks

- We just consider `Concrete` and `Bricks`
- Differential attacks:
  - Each active $x \mapsto x^2$ map has $\text{DP}_{\max} = 1/p$
    - Main issue: $x_0 \mapsto x_0$ in `Bricks`
  - We show that two consecutive rounds have DP$\leq p^{-t/8-1/2}$

  $\Rightarrow$ 6 rounds have DP$\leq 2^{-256}$ (for $t \geq 8$)
- Other attacks:
  - Rebound attack, truncated differential attacks, ...
  - Work in progress...

Security Analysis (cont.)

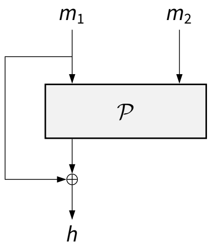
- Open points:
  - Full statistical analysis
  - Analysis/experiments for Gröbner basis attacks
- Preliminary Number of Rounds:

Security (bits)	r
80	7
128	**8**
196	10
256	12

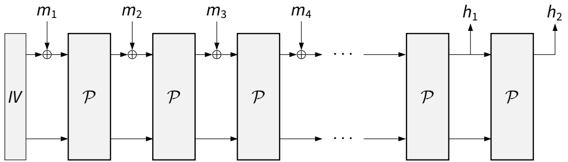
Performance

Modes of operation

- 2:1 compression:
  - $t = 8$ is sufficient!
  - E.g., for Merkle-tree with fixed depth



- General purpose hashing:
  - Use a sponge $t >= 12$
  - 4 words for capacity



Performance Summary

- Used prime field allows cheap/fast modular reduction
- Only $2t - 1$ modular reductions per round
  - Before `Bars`
  - After squares in `Bricks`
- `Bars` efficiently vectorizable without lookup tables
  - Cheap side channel resistant implementation possible
  - Use lookup tables in proof system
- `Concrete` chosen to minimize number of additions
  - No multiplications required!

Plain Performance (for 8 Rounds)

Table: Plain performance comparison implemented in Rust.

Hashing algorithm	Time (*ns*)		
	$t = 8$	$t = 12$	
$\texttt{RC}_p$	**147.6**	**237.5**	
Tip5 ($t = 16$)			487.0
Tip4′	-	252.0	
POSEIDON	2011.2	3510.5	
Poseidon2	973.0	1361.8	
`Reinforced Concrete` (BN254, $t = 3$)			1467.1
SHA3-256			189.8
SHA-256			45.3
`Concrete`	17.8	29.2	
`Bricks`	14.4	22.5	
`Bars`	12.2	16.9	

Constant Time Performance (for 8 Rounds)

- Resisting side-channel attacks is important, even in ZK use cases
  - E.g., recently shown at Usenix by [TBP20]
- Benchmarks when replacing fast modular reduction with constant time one:

Hashing algorithm	Time (*ns*) $t = 8$	Time (*ns*) $t = 12$
`RC`$_p$	**358.1**	**535.9**
Poseidon	4135.0	6960.4
Poseidon2	2011.0	2695.5
`Concrete`	34.6	50.1
`Bricks`	17.7	29.6
`Bars`	12.2	20.0

- Unrolling S-box for `Reinforced Concrete`, Tip5, Tip4$'$ likely very expensive

Proof System Performance - Plonkish

- `Bricks`:
  - $t-1$ polynomial constraints of degree 2
- `Bars`:
  - Decompostion: t linear constraints
  - $4t$ lookup constraints for $S(x)$
  - $2t$ polynomial constraints (degree 2) to ensure decompositions are $\in \mathbb{F}_p$
- Total for R rounds:
  - $4tR$ lookup constraints
  - tR linear constraints
  - $3tR$ polynomial constraints of degree 2

Proof System Performance - Plonkish (cont.)

- $\mathtt{RC}_p$ with 8 rounds for $t = 8$:
  - $32t = 256$ lookup constraints
  - $8t = 64$ linear constraints
  - $24t = 192$ polynomial constraints of degree 2

  $\Rightarrow \approx 64t = 512$ constraints of degree ≤ 2
- POSEIDON/Poseidon2 for $p = 2^{64} - 2^{32} + 1$ and $t = 8$
  - General: $t \cdot R_F + R_p - t + 1$ constraints of degree d
  - $7t + 23 = 79$ constraints of degree 7
  - Or: $28t + 92 = 316$ constraints of degree 2

$\Rightarrow$ $\mathtt{RC}_p$ has less degree-2 constraints, more in total

Conclusion

- New hash function $\texttt{RC}_p$
  - Efficient in plain and in proof systems
  - Plain performance faster than SHA-3!
  - Side-channel resistant and allows constant time implementations
- Design based on two different non-linear layers
  - `Bricks`: Arithmetic non-linear layer based on Feistel
  - `Bars`: Binary non-linear layer based on decomposition and χ
- Currently fastest arithmetization friendly hash function
- Generalized description for other primes in paper (to appear soon™)

Questions

?

RC$_p$:

Fast Arithmetization-Friendly Hashing

Lorenzo Grassi, Dmitry Khovratovich, Reinhard Lüftenegger,
Christian Rechberger, Markus Schofnegger, **Roman Walch**

23.04.2023

Bibliography I

[AAB+20] Abdelrahaman Aly, Tomer Ashur, Eli Ben-Sasson, Siemen Dhooghe, and Alan Szepieniec. **Design of Symmetric-Key Primitives for Advanced Cryptographic Protocols**. IACR Trans. Symmetric Cryptol. 2020.3 (2020), pp. 1–45.

[AGP+19] Martin R. Albrecht, Lorenzo Grassi, Léo Perrin, Sebastian Ramacher, Christian Rechberger, Dragos Rotaru, Arnab Roy, and Markus Schofnegger. **Feistel Structures for MPC, and More**. ESORICS (2). Vol. 11736. Lecture Notes in Computer Science. Springer, 2019, pp. 151–171.

[BBC+22] Clémence Bouvier, Pierre Briaud, Pyrros Chaidos, Léo Perrin, and Vesselin Velichkov. **Anemoi: Exploiting the Link between Arithmetization-Orientation and CCZ-Equivalence**. IACR Cryptol. ePrint Arch. (2022), p. 840.

[GHR+22] Lorenzo Grassi, Yonglin Hao, Christian Rechberger, Markus Schofnegger, Roman Walch, and Qingju Wang. **Horst Meets Fluid-SPN: Griffin for Zero-Knowledge Applications**. IACR Cryptol. ePrint Arch. (2022), p. 403.

Bibliography II

[GKL+22] Lorenzo Grassi, Dmitry Khovratovich, Reinhard Lüftenegger, Christian Rechberger, Markus Schofnegger, and Roman Walch. **Reinforced Concrete: A Fast Hash Function for Verifiable Computation.** CCS. ACM, 2022, pp. 1323–1335.

[GKR+21] Lorenzo Grassi, Dmitry Khovratovich, Christian Rechberger, Arnab Roy, and Markus Schofnegger. **Poseidon: A New Hash Function for Zero-Knowledge Proof Systems.** USENIX Security Symposium. USENIX Association, 2021, pp. 519–535.

[GKS23] Lorenzo Grassi, Dmitry Khovratovich, and Markus Schofnegger. **Poseidon2: A Faster Version of the Poseidon Hash Function.** IACR Cryptol. ePrint Arch. (2023), p. 323.

[GOPS22] Lorenzo Grassi, Silvia Onofri, Marco Pedicini, and Luca Sozzi. **Invertible Quadratic Non-Linear Layers for MPC-/FHE-/ZK-Friendly Schemes over Fnp Application to Poseidon.** IACR Trans. Symmetric Cryptol. 2022.3 (2022), pp. 20–72.

[Sal23] Robin Salen. **Two additional instantiations from the Tip5 hash function construction.** https://toposware.com/paper_tip5.pdf (2023).

Bibliography III

[SLST23] Alan Szepieniec, Alexander Lemmens, Jan Ferdinand Sauer, and Bobbin Threadbare. **The Tip5 Hash Function for Recursive STARKs.** IACR Cryptol. ePrint Arch. (2023), p. 107.

[TBP20] Florian Tramèr, Dan Boneh, and Kenny Paterson. **Remote Side-Channel Attacks on Anonymous Transactions.** USENIX Security Symposium. USENIX Association, 2020, pp. 2739–2756.