Symmetric Primitives for MPC-/ZK-/HE-Applications

Lorenzo Grassi

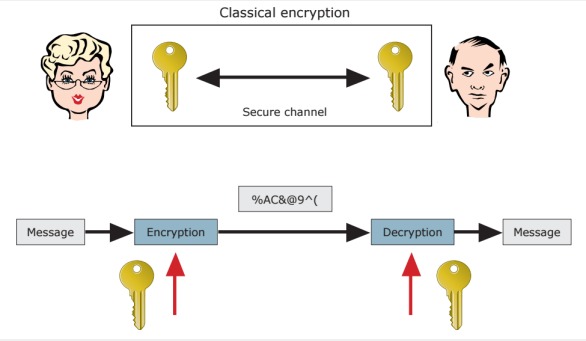
Eindhoven University of Technology, NL

May 2026

Table of Contents

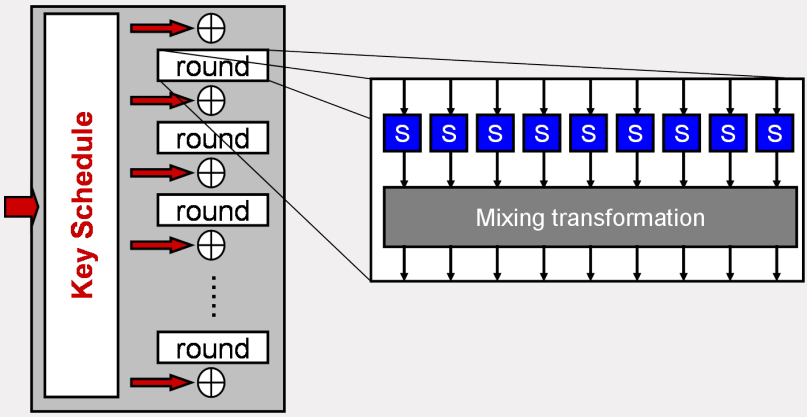
Table of Contents

Symmetric Cryptography (Confidentiality)



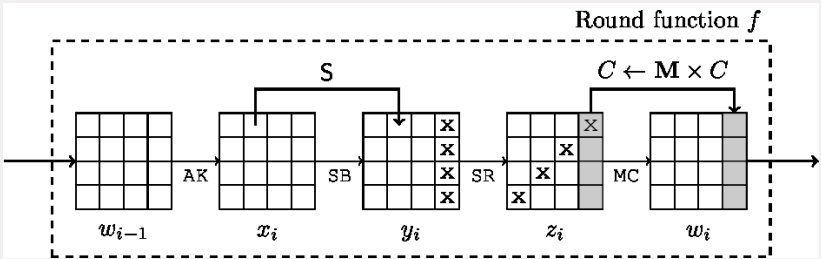
Reprinted from https://www.cosic.esat.kuleuven.be/summer_school_sardinia_2015/slides/LRKnudsen.pdf by *Lars R. Knudsen*

SPN Scheme


Key Schedule
round
round
round
round
S
S
S
S
S
S
S
S
S
Mixing transformation

Advance Encryption Scheme (AES)

- SPN block cipher based on the *Wide-Trail design strategy*;
- Block size of 128 bits = 16 bytes, organized in a 4×4 matrix;
- Key size of 128/192/256 bits;
- 10/12/14 rounds $x \mapsto k^i \oplus \text{MC} \circ \text{SR} \circ \text{S-Box}(x)$:



Security of Ciphers (and Symmetric Primitives)

When is a cipher secure?

Kerckhoffs' Principle: the security of a cryptosystem must lie in the choice of its keys only. Everything else (including the algorithm itself) should be considered public knowledge.

A symmetric primitive is secure if there is no attack better than brute force: a solid symmetric primitive must resist *all* known attacks published in the literature!

Security of Ciphers (and Symmetric Primitives)

When is a cipher secure?

Kerckhoffs' Principle: the security of a cryptosystem must lie in the choice of its keys only. Everything else (including the algorithm itself) should be considered public knowledge.

A symmetric primitive is secure if there is no attack better than brute force: a solid symmetric primitive must resist *all* known attacks published in the literature!

Security of Ciphers (and Symmetric Primitives)

When is a cipher secure?

Kerckhoffs' Principle: the security of a cryptosystem must lie in the choice of its keys only. Everything else (including the algorithm itself) should be considered public knowledge.

A symmetric primitive is secure if there is no attack better than brute force: a solid symmetric primitive must resist *all* known attacks published in the literature!

Cryptographic Attacks

- *Statistical Attacks:* attacks that exploit one or more properties of the attacked primitive that hold(s) with certain probability.

  Examples: differential cryptanalysis, linear cryptanalysis, integral attack, and their variants.

- *Algebraic Attacks:* attacks that exploit the simple algebraic description of the attacked primitive.

  Examples: interpolation attack, Gröbner basis, higher-order differential, and more.

Cryptographic Attacks

- *Statistical Attacks:* attacks that exploit one or more properties of the attacked primitive that hold(s) with certain probability.

  Examples: differential cryptanalysis, linear cryptanalysis, integral attack, and their variants.

- *Algebraic Attacks:* attacks that exploit the simple algebraic description of the attacked primitive.

  Examples: interpolation attack, Gröbner basis, higher-order differential, and more.

Hash Functions

A cryptographical secure hash function H is a (deterministic) one-way function

$$H : \mathbb{F}^* \to \mathbb{F}^\eta$$

with the following properties:

- *pre-image resistance:* given h, hard to find x s.t. $H(x) = h$;
- *second pre-image resistance:* given $H(x) = h$, hard to find $x' \neq x$ s.t. $H(x') = h$;
- *collision resistance:* hard to find $x' \neq x$ s.t. $H(x) = H(x')$.

Examples: SHA-2 (Merkle-Damgård), Keccak/SHA-3 (Sponge), ...

Table of Contents

Recent Applications

Symmetric cryptography primitives may be needed in:

- secure multi-party computation (MPC),
- homomorphic encryption (HE),
- zero-knowledge proofs (ZK),

where

1. *details of the used symmetric algorithm may influence the protocols efficiency;*
2. *many of such protocols are naturally defined over* $(\mathbb{F}_p)^n$ *for a large prime integer* p (e.g., $p \approx 2^{32}, 2^{64}, 2^{128}$, or 2^{256}).

Recent Applications

Symmetric cryptography primitives may be needed in:

- secure multi-party computation (MPC),
- homomorphic encryption (HE),
- zero-knowledge proofs (ZK),

where

1. *details of the used symmetric algorithm may influence the protocols efficiency*;
2. *many of such protocols are naturally defined over* $(\mathbb{F}_p)^n$ *for a large prime integer* p (e.g., $p \approx 2^{32}, 2^{64}, 2^{128}$, or 2^{256}).

About MPC, ZK, and HE

- MPC: joint evaluation a function on private inputs:
  - input: parties with (private) input x_i for $i \in \{1, 2, \ldots, n\}$;
  - output: jointly compute a (known) function $y = F(x_1, \ldots, x_n)$ such that *correctness* and *privacy* are guaranteed.

- ZK: method by which one party (the prover) can prove to another party (the verifier) that a given statement is true, without reveling any information beyond the mere fact of the statement's truth.

- HE: form of encryption that allows computations to be performed on encrypted data without first having to decrypt it:

$$\text{(private) input: } x \quad \rightarrow \quad \text{output: } F(x) \equiv E_k^{-1}\left(F'(E_k(x))\right) .$$

About MPC, ZK, and HE

- MPC: joint evaluation a function on private inputs:
  - input: parties with (private) input x_i for $i \in \{1, 2, \ldots, n\}$;
  - output: jointly compute a (known) function $y = F(x_1, \ldots, x_n)$ such that *correctness* and *privacy* are guaranteed.

- ZK: method by which one party (the prover) can prove to another party (the verifier) that a given statement is true, without reveling any information beyond the mere fact of the statement's truth.

- HE: form of encryption that allows computations to be performed on encrypted data without first having to decrypt it:

$$\text{(private) input: } x \quad \rightarrow \quad \text{output: } F(x) \equiv E_k^{-1}\left(F'(E_k(x))\right) .$$

About MPC, ZK, and HE

- MPC: joint evaluation a function on private inputs:
  - input: parties with (private) input x_i for $i \in \{1, 2, \ldots, n\}$;
  - output: jointly compute a (known) function $y = F(x_1, \ldots, x_n)$ such that *correctness* and *privacy* are guaranteed.

- ZK: method by which one party (the prover) can prove to another party (the verifier) that a given statement is true, without reveling any information beyond the mere fact of the statement's truth.

- HE: form of encryption that allows computations to be performed on encrypted data without first having to decrypt it:

$$\text{(private) input: } x \quad \rightarrow \quad \text{output: } F(x) \equiv E_k^{-1}\left(F'(E_k(x))\right) .$$

Cost Metric of MPC-/HE-/ZK-Friendly Schemes

Demand of new symmetric primitives over *prime fields* for these new applications!

Rough Cost Metric:

- Linear/Affine functions: *almost free*;
- Non-linear functions: *expensive*.

(*Important:* the size p of the field does not impact the cost in these MPC/HE/ZK applications!)

Cost Metric of MPC-/HE-/ZK-Friendly Schemes

Demand of new symmetric primitives over *prime fields* for these new applications!

Rough Cost Metric:

- Linear/Affine functions: *almost free*;
- Non-linear functions: *expensive*.

(*Important:* the size p of the field does not impact the cost in these MPC/HE/ZK applications!)

Cost Metric of MPC-/HE-/ZK-Friendly Schemes

Demand of new symmetric primitives over *prime fields* for these new applications!

Rough Cost Metric:

- Linear/Affine functions: *almost free*;
- Non-linear functions: *expensive*.

(*Important:* the size p of the field does not impact the cost in these MPC/HE/ZK applications!)

Cost Metrics for MPC

MPC:

- *multiplications require communication between the parties* $\Rightarrow$ total number of multiplications as a good estimation of the complexity of a MPC-friendly scheme;
- affine operations for “free” since they can be computed locally (but they influence the plain performance);
- other factors may influence the overall cost (e.g., number of offline/online communication rounds).

Cost Metrics for ZK (1/2)

ZK (R1CS and AIR):

- number of multiplications required during the verification process as a good estimation of the complexity of a ZK-friendly scheme;
- about affine (especially, linear) operations:
  - R1CS: almost for free;
  - AIR: they impact the final performance, but still cheaper than multiplications.

In Plonkup (Plonk + Plookup) and Binius:

- look-up tables are possible and relatively cheap $\rightarrow$ different cost metric!

Cost Metrics for ZK (1/2)

ZK (R1CS and AIR):

- number of multiplications required during the verification process as a good estimation of the complexity of a ZK-friendly scheme;
- about affine (especially, linear) operations:
  - R1CS: almost for free;
  - AIR: they impact the final performance, but still cheaper than multiplications.

In Plonkup (Plonk + Plookup) and Binius:

- look-up tables are possible and relatively cheap $\rightarrow$ different cost metric!

Cost Metrics for ZK – Examples (2/2)

Given x and $y = x^{p-2} \equiv 1/x$ over $\mathbb{F}_p$, verified via

$$\forall x, y \neq 0 : \qquad x \cdot y = 1 \,.$$

Given x and $y = x^{1/d}$ over $\mathbb{F}_p$ s.t. $\gcd(d, p-1) = 1$, then verified via

$$y^d - x = 0 \,.$$

(*Note:* if d is small, then $1/d$ is huge! E.g., $d = 3$ and $1/d = (2p-1)/3$.)

Cost Metrics for ZK – Examples (2/2)

Given x and $y = x^{p-2} \equiv 1/x$ over $\mathbb{F}_p$, verified via

$$\forall x, y \neq 0 : \qquad x \cdot y = 1 \,.$$

Given x and $y = x^{1/d}$ over $\mathbb{F}_p$ s.t. $\gcd(d, p-1) = 1$, then verified via

$$y^d - x = 0 \,.$$

(*Note:* if d is small, then $1/d$ is huge! E.g., $d = 3$ and $1/d = (2p-1)/3$.)

Cost Metrics for HE

HE:

- multiplicative depth of the circuit/scheme to evaluate as a good estimation of the complexity of HE-friendly scheme (e.g., for *reducing the ciphertext expansion* and so bootstrapping);
- number of multiplications: it does not impact the performance;
- about affine operations: almost for free.

Research of “New” Symmetric Primitives

“Traditional” symmetric primitives (e.g., AES, Keccak, ...) designed for being efficient in Hardware/Software (HW/SW) $\Rightarrow$ not suitable for MPC/HE/ZK applications!

1. *Designer Approach:*
   - “Traditional” primitives: defined over *small binary* fields $\mathbb{F}_{2^n}^t$ with $n \in \{3, 4, \ldots, 8\}$ for which cost XOR $\approx$ cost AND;
   - MPC-/HE-/ZK-primitives: defined over *large prime* fields $\mathbb{F}_p^t$ (i.e., $\log_2(p) \approx 32, 64, 128$ or 256) with *minimal multiplicative complexity*;
2. *Attacker Approach:*
   - “Traditional” primitives: mainly, statistical attacks (differential, linear, ...);
   - MPC-/HE-/ZK-primitives: mainly, algebraic attacks (Gröbner basis, ...), as *non-linear layers defined by simple algebraic expression*.

Research of "New" Symmetric Primitives

"Traditional" symmetric primitives (e.g., AES, Keccak, ...) designed for being efficient in Hardware/Software (HW/SW) $\Rightarrow$ not suitable for MPC/HE/ZK applications!

1. *Designer Approach:*
   - "Traditional" primitives: defined over *small binary* fields $\mathbb{F}_{2^n}^t$ with $n \in \{3, 4, \ldots, 8\}$ for which cost XOR $\approx$ cost AND;
   - MPC-/HE-/ZK-primitives: defined over *large prime* fields $\mathbb{F}_p^t$ (i.e., $\log_2(p) \approx 32, 64, 128$ or 256) with *minimal multiplicative complexity*;
2. *Attacker Approach:*
   - "Traditional" primitives: mainly, statistical attacks (differential, linear, ...);
   - MPC-/HE-/ZK-primitives: mainly, algebraic attacks (Gröbner basis, ...), as *non-linear layers defined by simple algebraic expression.*

Research of "New" Symmetric Primitives

"Traditional" symmetric primitives (e.g., AES, Keccak, ...) designed for being efficient in Hardware/Software (HW/SW) $\Rightarrow$ not suitable for MPC/HE/ZK applications!

1. *Designer Approach:*
   - "Traditional" primitives: defined over *small binary* fields $\mathbb{F}_{2^n}^t$ with $n \in \{3, 4, \ldots, 8\}$ for which cost XOR $\approx$ cost AND;
   - MPC-/HE-/ZK-primitives: defined over *large prime* fields $\mathbb{F}_p^t$ (i.e., $\log_2(p) \approx 32, 64, 128$ or 256) with *minimal multiplicative complexity*;
2. *Attacker Approach:*
   - "Traditional" primitives: mainly, statistical attacks (differential, linear, ...);
   - MPC-/HE-/ZK-primitives: mainly, algebraic attacks (Gröbner basis, ...), as *non-linear layers defined by simple algebraic expression*.

How Many MPC/-HE/-ZK-friendly symmetric primitives?

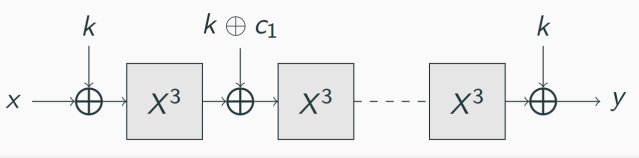
- $\leq$ 2014: -
- 2015: 1
- 2016: 4
- 2017: -
- 2018: 3
- 2019: 5
- 2020: 5
- 2021: 8
- 2022: 10

Table of Contents

The MPC-friendly Symmetric Crypto Zoo

- 2015: LowMC
- 2016: MiMC, LegendrePRF
- 2017: –
- 2018: CryptoDarkMatter
- 2019: GMiMC
- 2020: HadesMiMC
- 2021: Ciminion, "CryptoDarkMatter++"
- 2022: –
- 2023: Hydra, MiMC++, Pluto

MiMC Cipher

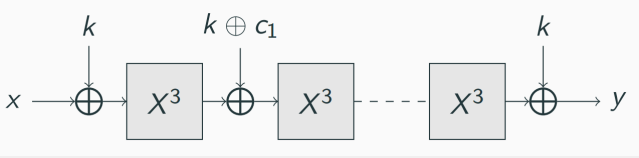


($x \mapsto x^3$ is a permutation **iff** $n = 2n' + 1$ odd and $p \equiv_3 2$)

Assuming $p \approx 2^n$, *large number of rounds*: $\lceil \log_3 p \rceil \approx \lceil n \cdot \log_3 2 \rceil$.
E.g., for $p \approx 2^{128}$:

- AES: 10 rounds and $\approx$ 960 (MPC) multiplications;
- MiMC: 81 rounds and 162 (MPC) multiplications.

MiMC Cipher

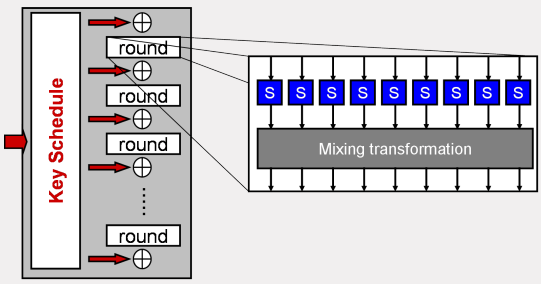


($x \mapsto x^3$ is a permutation **iff** $n = 2n' + 1$ odd and $p \equiv_3 2$)

Assuming $p \approx 2^n$, *large number of rounds*: $\lceil \log_3 p \rceil \approx \lceil n \cdot \log_3 2 \rceil$.
E.g., for $p \approx 2^{128}$:

- AES: 10 rounds and $\approx$ 960 (MPC) multiplications;
- MiMC: 81 rounds and 162 (MPC) multiplications.

Partial-SPN Symmetric Primitives



Idea: Move from full S-Box layer

$$\mathcal{S}_F(x) = [S(x_1)||S(x_2)|| \ldots ||S(x_t)]$$

to Partial S-Box layer

$$\mathcal{S}_P(x) = [S(x_1)||x_2|| \ldots ||x_t]\,.$$

P-SPN versus SPN: Advantages and Disadvantages

Advantages of P-SPN:

- cheaper to compute than SPN
- one S-Box per round is sufficient for increasing the overall degree, crucial for preventing algebraic attacks (including Gröbner Basis);

but

- guarantee security of P-SPN against statistical attacks is harder than for SPN: the *"wide-trail" design strategy* (= choose linear layers that active as many S-Boxes as possible over multiple rounds – used in the AES) does not apply. Ad-hoc security argument must be provided.

Examples: attacks against the P-SPN schemes Zorro (variant of AES) and LowMC.

P-SPN versus SPN: Advantages and Disadvantages

Advantages of P-SPN:

- cheaper to compute than SPN
- one S-Box per round is sufficient for increasing the overall degree, crucial for preventing algebraic attacks (including Gröbner Basis);

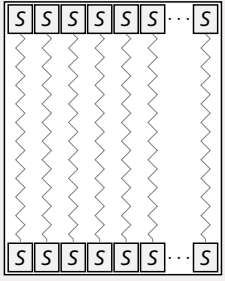
but

- guarantee security of P-SPN against statistical attacks is harder than for SPN: the *"wide-trail" design strategy* (= choose linear layers that active as many S-Boxes as possible over multiple rounds – used in the AES) does not apply. Ad-hoc security argument must be provided.

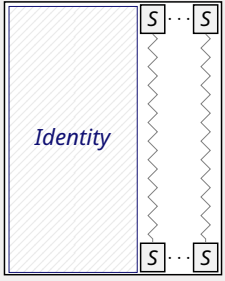
Examples: attacks against the P-SPN schemes Zorro (variant of AES) and LowMC.

“Hades” Strategy

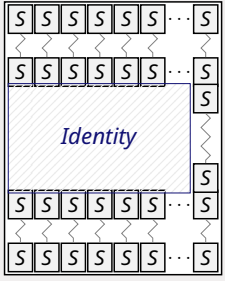
How to reduce number of non-linear operations & guarantee security with simple/elegant argument?



(a) SPN

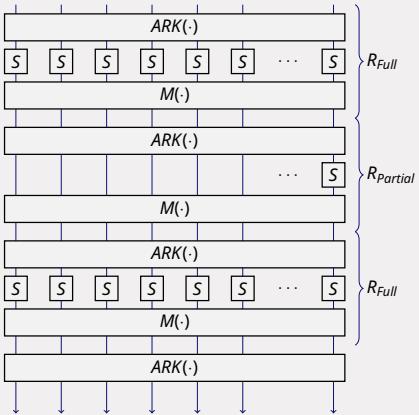


(b) P-SPN



(c) *“Hades” strategy*

The Block Cipher HadesMiMC



- $S(x) = x^d$ where $\gcd(d, p-1) = 1$;
- Linear layer: multiplication with a MDS matrix in $\mathbb{F}_p^{t \times t}$;
- Subkeys defined via an affine map applied to the master key;
- Number of rounds ($\kappa \approx \log_2(p)$):

$$R_F = 2 \cdot R_f = 6\,,$$
$$R_P \approx \log_d(p)$$

Number Multiplications of MPC-Friendly Schemes

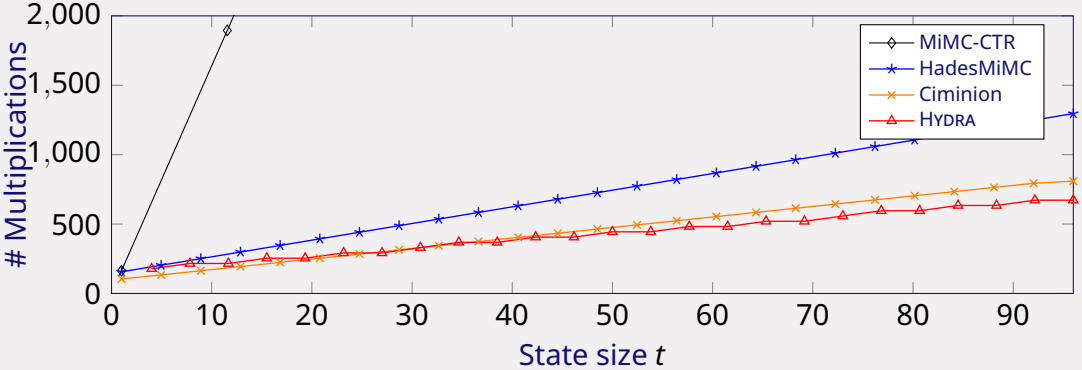


Figure: Number of multiplications of several MPC-friendly designs over $\mathbb{F}_p^t$, with $p \approx 2^{128}$ and $t \geq 2$ (security level of 128 bits).

Table of Contents

The ZK-friendly Symmetric Crypto Zoo (1/2)

- 2018: Friday/Jarvis
- 2019: (Sponge Hash) GMiMC
- 2020: Vision/Rescue, Grendel
- 2021: Poseidon
- 2022: `Reinforced Concrete`, Neptune
- 2023: Griffin, Anemoi, Poseidon2

The ZK-friendly Symmetric Crypto Zoo (2/2)

Type 1

- Low-degree

$$y = x^d$$

- Fast in Plain
- Many rounds
- Often more constraints
- GMiMC, Poseidon, Neptune, Poseidon2, ...

Type 2

- Low-degree equivalence

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

- Slow in Plain
- Fewer rounds
- Fewer constraints
- Vision, Rescue, Grendel, Griffin, Anemoi, Arion, ...

Type 3

- Lookup tables

$$y = \mathcal{T}[x]$$

- Fast in Plain
- Fewer rounds
- Constraints depend on proof system
- Reinforced Concrete, Tip5, Skyscraper, ...

The ZK-friendly Symmetric Crypto Zoo (2/2)

Type 1

- Low-degree

$$y = x^d$$

- Fast in Plain
- Many rounds
- Often more constraints
- GMiMC, Poseidon, Neptune, Poseidon2, ...

Type 2

- Low-degree equivalence

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

- Slow in Plain
- Fewer rounds
- Fewer constraints
- Vision, Rescue, Grendel, Griffin, Anemoi, Arion, ...

Type 3

- Lookup tables

$$y = \mathcal{T}[x]$$

- Fast in Plain
- Fewer rounds
- Constraints depend on proof system
- Reinforced Concrete, Tip5, Skyscraper, ...

The ZK-friendly Symmetric Crypto Zoo (2/2)

Type 1

- Low-degree

$$y = x^d$$

- Fast in Plain
- Many rounds
- Often more constraints
- GMiMC, Poseidon, Neptune, Poseidon2, …

Type 2

- Low-degree equivalence

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

- Slow in Plain
- Fewer rounds
- Fewer constraints
- Vision, Rescue, Grendel, Griffin, Anemoi, Arion, …

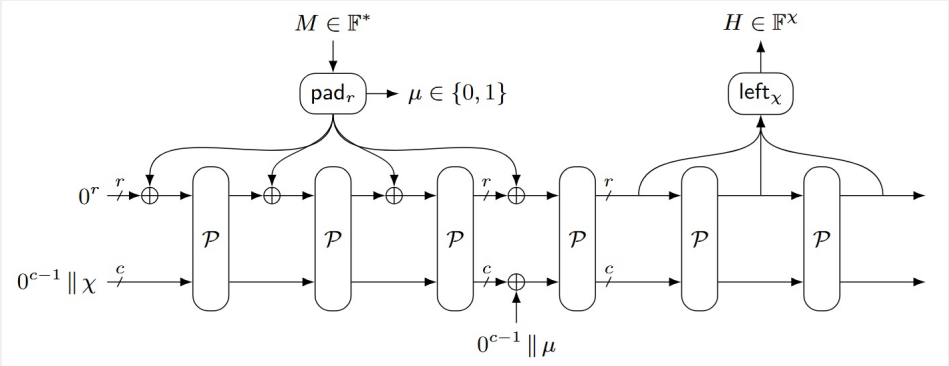
Type 3

- Lookup tables

$$y = \mathcal{T}[x]$$

- Fast in Plain
- Fewer rounds
- Constraints depend on proof system
- `Reinforced Concrete`, Tip5, `Skyscraper`, …

Sponge Hash Function



Assuming $\mathcal{P}$ over $\mathbb{F}_2^{c+r}$ is an ideal permutation: security up to $\min\{2^\chi, 2^{c/2}\}$.

TU/e

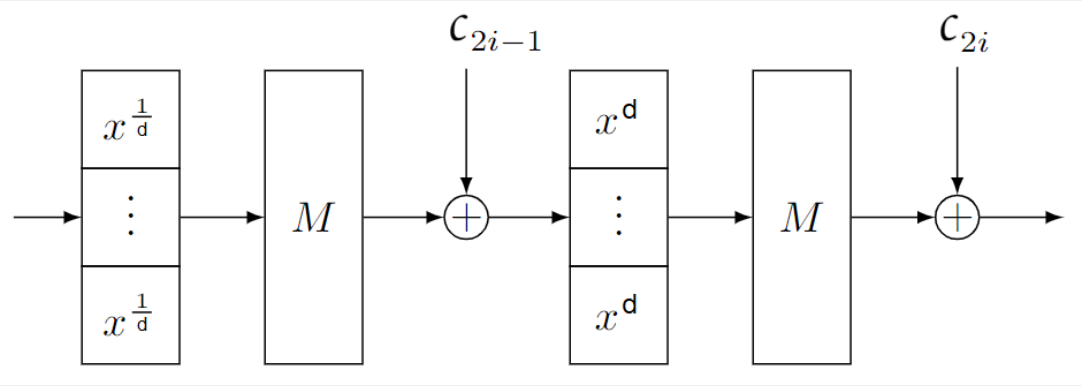
Poseidon

- Poseidon is a sponge hash function instantiated by the HadesMiMC permutation (that is, round keys are replaced by round constants).
- Number of rounds for Poseidon is a bit different than the number of rounds of HadesMiMC (due to different attacks):
  - $4 + 4 = 8$ external full rounds (instead of 6);
  - partial rounds still $\approx \log_d(p)$.
- Low degree permutation: used both for evaluation and for verification.

Poseidon2

Same number of rounds of Poseidon, but (i) two different linear layers (one for external rounds & one for internal ones) + (ii) additional initial linear layer:

Rescue



Rescue: Security and Verification

- Within few rounds: high (maximum) degree + density of the interpolation polynomial $\Rightarrow$ security against *algebraic attacks* in few rounds!
- Security against *statistical attacks* via the “wide-trail” design strategy.
- Efficient verification, as given $y = R(x) \in \mathbb{F}_p^t$:

$$\forall i: \quad \sum_j M_{i,j} \cdot x_j^d = \left(\sum_l M^{-1}{}_{i,l} \cdot y_l \right)^d .$$

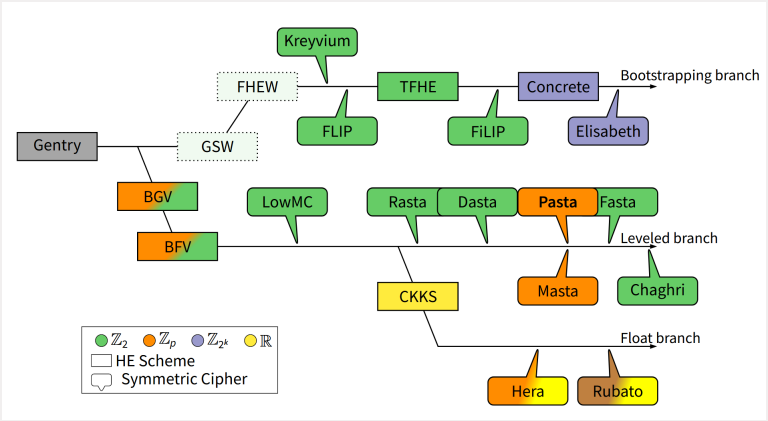
Rescue: Security and Verification

- Within few rounds: high (maximum) degree + density of the interpolation polynomial $\Rightarrow$ security against *algebraic attacks* in few rounds!
- Security against *statistical attacks* via the “wide-trail” design strategy.
- Efficient verification, as given $y = R(x) \in \mathbb{F}_p^t$:

$$\forall i: \quad \sum_j M_{i,j} \cdot x_j^d = \left(\sum_l M^{-1}{}_{i,l} \cdot y_l \right)^d .$$

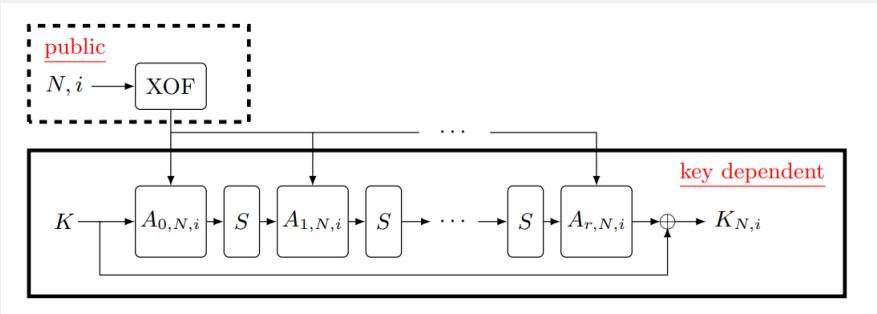
Table of Contents

The HE-friendly Symmetric Crypto Zoo



Reprinted from "On FHE/MPC/ZK-friendly symmetric crypto" by Christian Rechberger (TU Graz)

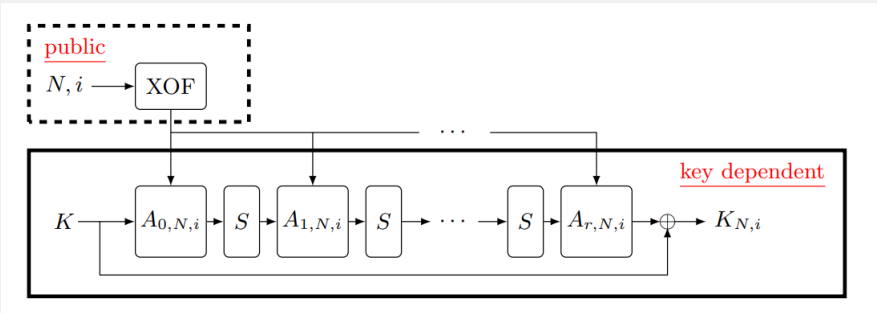
Rasta (over $\mathbb{F}_2^n$)



Non-linear layer S instantiated by the chi-function $x \mapsto x \oplus (\overline{x} \lll 1) \cdot (x \lll 2)$:

- degree 2 per round, hence depth 1;
- total depth (= number of rounds): between 4 and 6.

Rasta (over $\mathbb{F}_2^n$)



Non-linear layer S instantiated by the chi-function $x \mapsto x \oplus (\bar{x} \lll 1) \cdot (x \lll 2)$:

- degree 2 per round, hence depth 1;
- total depth (= number of rounds): between 4 and 6.

Security of Rasta – Statistical Attacks

Goal: keep the depth as smaller as possible!

- Cryptographic attacks assume that the ciphertexts are generated by a fixed scheme (that is, fixed S-Box, fixed linear layer, ...).
- **Idea:** change some components of the scheme at every encryption $\Rightarrow$ since ciphertexts are generated w.r.t. different schemes, statistical attacks (as differential and linear ones) do not work!
- Due to the cost metric, *each affine layer can be randomly generated at every new encryption without affecting the overall cost!*

Security of Rasta – Statistical Attacks

Goal: keep the depth as smaller as possible!

- Cryptographic attacks assume that the ciphertexts are generated by a fixed scheme (that is, fixed S-Box, fixed linear layer, ...).
- **Idea:** change some components of the scheme at every encryption $\Rightarrow$ since ciphertexts are generated w.r.t. different schemes, statistical attacks (as differential and linear ones) do not work!
- Due to the cost metric, *each affine layer can be randomly generated at every new encryption without affecting the overall cost!*

Security of Rasta – Statistical Attacks

Goal: keep the depth as smaller as possible!

- Cryptographic attacks assume that the ciphertexts are generated by a fixed scheme (that is, fixed S-Box, fixed linear layer, ...).
- **Idea:** change some components of the scheme at every encryption $\Rightarrow$ since ciphertexts are generated w.r.t. different schemes, statistical attacks (as differential and linear ones) do not work!
- Due to the cost metric, *each affine layer can be randomly generated at every new encryption without affecting the overall cost!*

Security of Rasta – Statistical Attacks

Goal: keep the depth as smaller as possible!

- Cryptographic attacks assume that the ciphertexts are generated by a fixed scheme (that is, fixed S-Box, fixed linear layer, ...).
- **Idea:** change some components of the scheme at every encryption $\Rightarrow$ since ciphertexts are generated w.r.t. different schemes, statistical attacks (as differential and linear ones) do not work!
- Due to the cost metric, *each affine layer can be randomly generated at every new encryption without affecting the overall cost!*

Security of Rasta – Algebraic Attacks

- Previous strategy does **not** impact some algebraic attacks.
  E.g., Gröbner basis (GB) attacks:
  - for a single input/output, set up the system of equations;
  - solve it, and find the key!
- **Idea:** since only care about the depth, we increase the number of variables in the system, making GB (and other algebraic) attacks infeasible!
- Examples (for 128 bits of security):
  - depth (=rounds) 3: state size $\approx 2^{18}$ bits;
  - depth (=rounds) 4: state size = 1877 bits;
  - depth (=rounds) 5: state size = 525 bits;
  - depth (=rounds) 6: state size = 351 bits.

Security of Rasta – Algebraic Attacks

- Previous strategy does **not** impact some algebraic attacks.
  E.g., Gröbner basis (GB) attacks:
  - for a single input/output, set up the system of equations;
  - solve it, and find the key!
- **Idea:** since only care about the depth, we increase the number of variables in the system, making GB (and other algebraic) attacks infeasible!
- Examples (for 128 bits of security):
  - depth (=rounds) 3: state size $\approx 2^{18}$ bits;
  - depth (=rounds) 4: state size = 1877 bits;
  - depth (=rounds) 5: state size = 525 bits;
  - depth (=rounds) 6: state size = 351 bits.

Table of Contents

Summary

- Lots of exciting new developments in "high functionality cryptography" – some are likely here to stay
- … leading to lots of exciting research for design and analysis of symmetric crypto and hashing!
- Industry interest is growing, demand for standards to support interoperability and increase trust.

Thanks for your attention!

Questions?

Comments?