Symmetric Techniques for Advanced Protocols: What *are* They?

Léo Perrin[1]

Inria, Paris

14th of March 2025

Trendy topics

MPC-friendly?

Arithmetization-Oriented?

Verification efficiency?

Algebraic attacks?

Trendy topics

MPC-friendly?

Arithmetization-Oriented?

Verification efficiency?

Algebraic attacks?

Symmetric crypto **for the blockchain...**

Trendy topics

MPC-friendly?

Arithmetization-Oriented?

Verification efficiency?

Algebraic attacks?

Symmetric crypto **for the blockchain...**

... for neural networks???

The conclusion of today: **symmetric cryptography** has always had to deal with specific **implementation criteria**, but the new ones are indeed a bit **stranger than before**.

Outline

1. What is the Purpose of a Symmetric Primitive
2. “Advanced” Protocols
3. Symmetric Primitives for Advanced Protocols
4. Conclusion

Plan of this Section

1 What is the Purpose of a Symmetric Primitive

2 "Advanced" Protocols

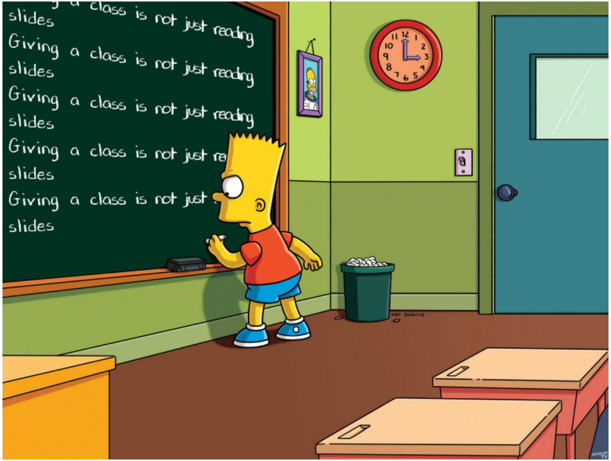
3 Symmetric Primitives for Advanced Protocols

4 Conclusion

Plan of this Section

1. What is the Purpose of a Symmetric Primitive
   - Let’s look at primitives we all know
   - A Small Cog in a Big Machine
2. “Advanced” Protocols
3. Symmetric Primitives for Advanced Protocols
4. Conclusion

Why do we need symmetric primitives?


Giving a class is not just reading slides
Giving a class is not just reading slides
Giving a class is not just reading slides
Giving a class is not just rea
slides
Giving a class is not just
slides

Unstable Definitions

What is "efficient" varies

- What are the operations that we **can** use?
- What are the associated **costs**?

How to get the best security for a given price?

What is "secure" varies

- Should the primitive work in many context?
- Do we care about nonce-misuse?

How do we define the **security** that the primitive must provide?

Unstable Definitions

What is "efficient" varies

- What are the operations that we **can** use?
- What are the associated **costs**?

How to get the best security for a given price?

What is "secure" varies

- Should the primitive work in many context?
- Do we care about nonce-misuse?

How do we define the **security** that the primitive must provide?

What are the relevant forms of cryptanalysis?

Unstable Definitions

What is “efficient” varies

- What are the operations that we **can** use?
- What are the associated **costs**?

How to get the best security for a given price?

What is “secure” varies

- Should the primitive work in many context?
- Do we care about nonce-misuse?

How do we define the **security** that the primitive must provide?

What are the relevant forms of cryptanalysis?

Unstable Definitions

What is "efficient" varies

- What are the operations that we **can** use?
- What are the associated **costs**?

How to get the best security for a given price?

What is "secure" varies

- Should the primitive work in many context? Modularity vs. Single use
- Do we care about nonce-misuse? Robustness vs. "not our problem"

How do we define the **security** that the primitive must provide?

What are the relevant forms of cryptanalysis?

Plan of this Section

1 What is the Purpose of a Symmetric Primitive
- Let's look at primitives we all know
- A Small Cog in a Big Machine

2 "Advanced" Protocols

3 Symmetric Primitives for Advanced Protocols

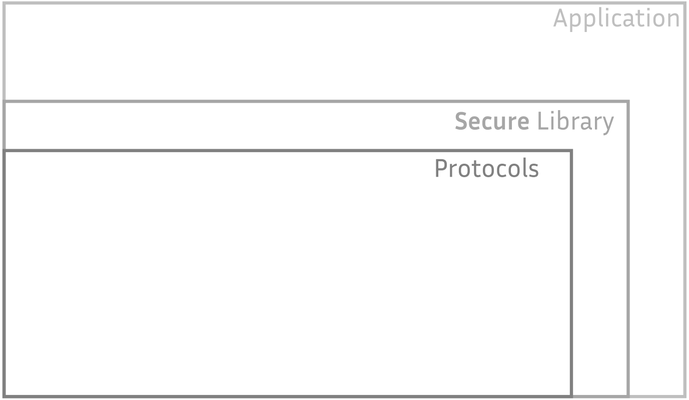
4 Conclusion

Web Encryption

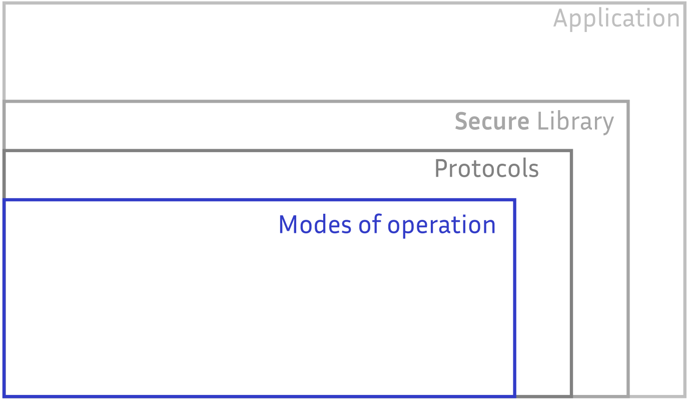
Web Encryption

Application
Secure Library

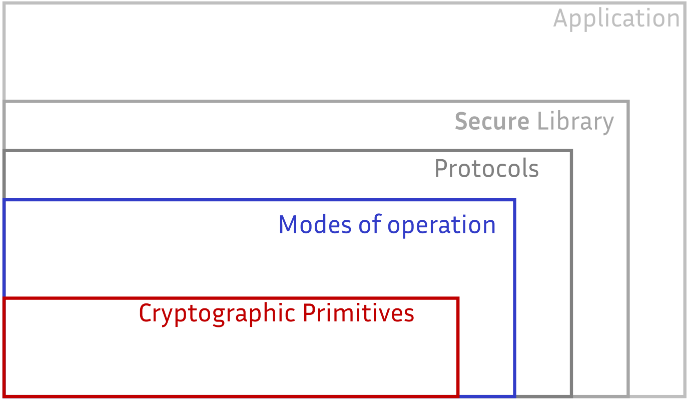
Web Encryption



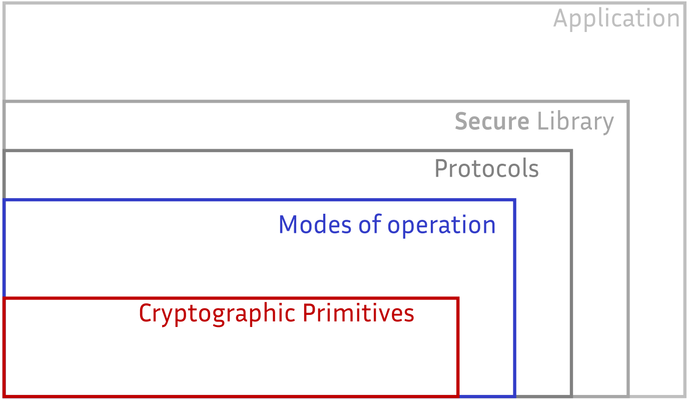
Web Encryption



Web Encryption

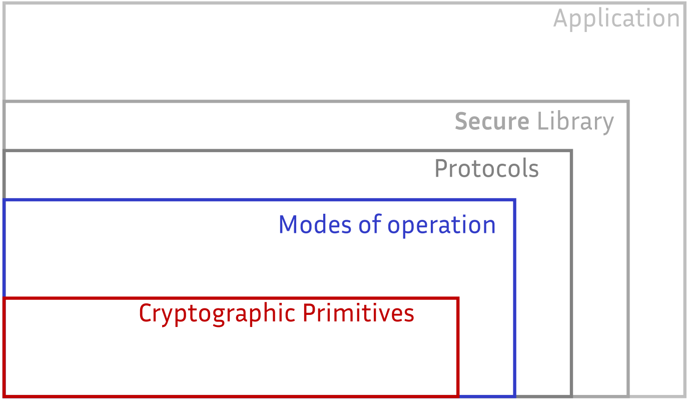


Web Encryption



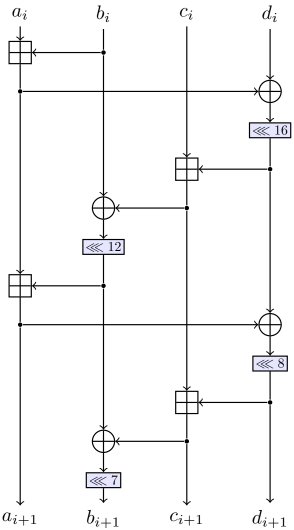
- We want software efficient (computer and smartphone but not micro-controllers) efficient AEAD for packets of a few tens to a few billion bytes.

Web Encryption



- We want software efficient (computer and smartphone but not micro-controllers) efficient AEAD for packets of a few tens to a few billion bytes.
- AES-GCM; Chacha-poly1305.

What Chacha looks like



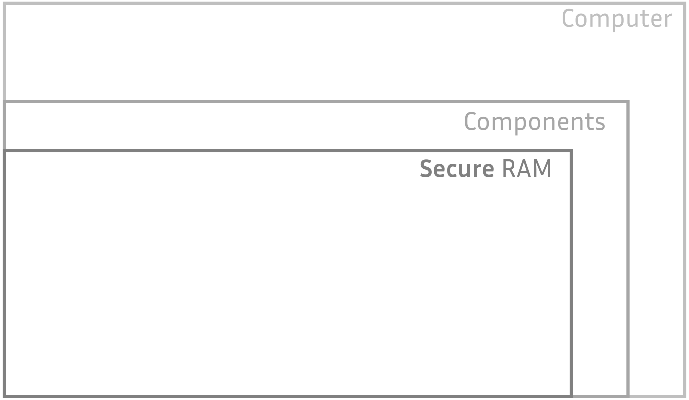
- Addition / Rotation / XOR
- 256-bit key
- 512-bit state
- Defined over 32-bit words

RAM Encryption

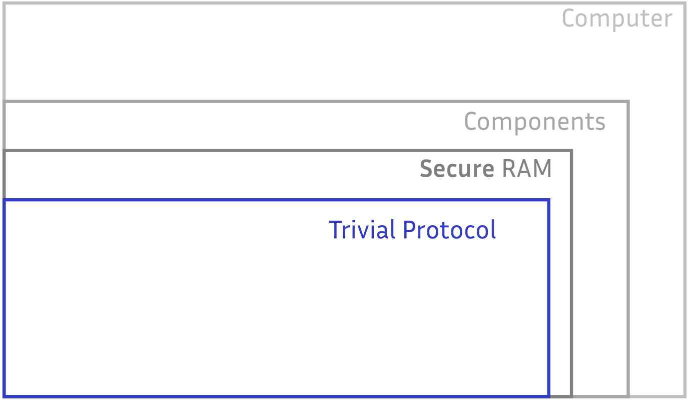
Computer

RAM Encryption

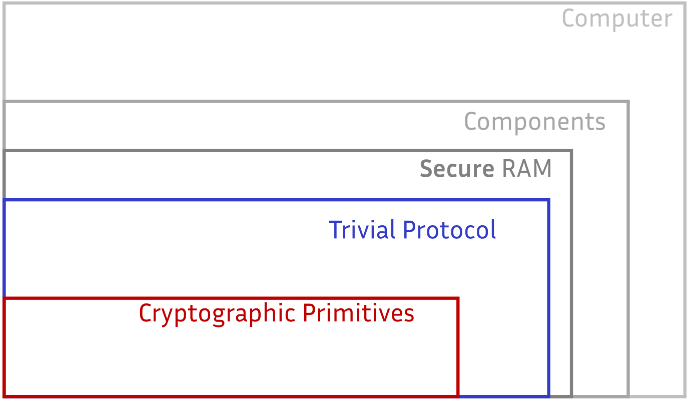
RAM Encryption



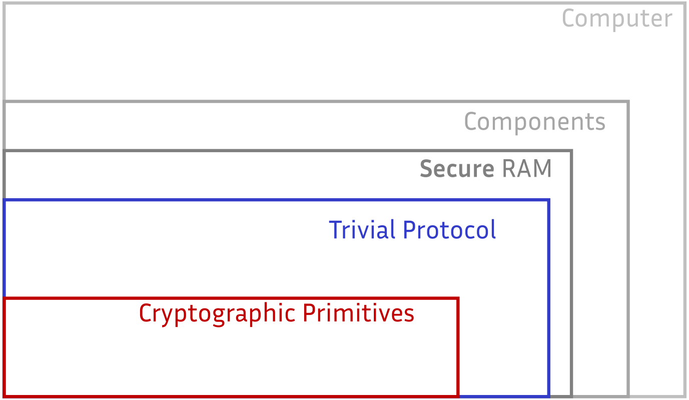
RAM Encryption



RAM Encryption

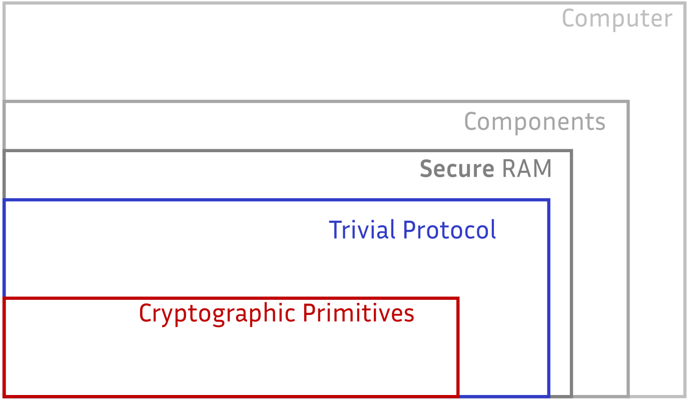


RAM Encryption



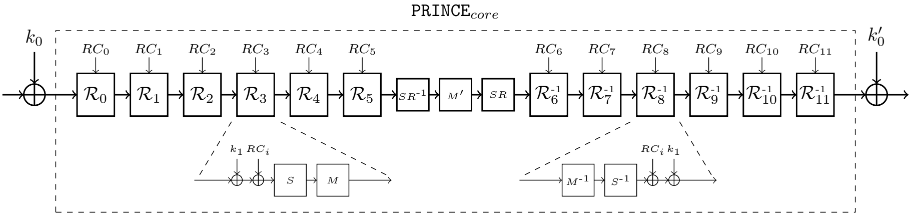
- We want very low latency block encryption for specific (and small) block sizes.

RAM Encryption



- We want very low latency block encryption for specific (and small) block sizes.
- PRINCE? QARMA? not so clear at this stage.

What PRINCE looks like



- 4-bit S-box optimized for hardware
- 2 different matrices
- FX construction
- “α-reflexion”
- inverse rounds used in the second half

Some Constants

- A symmetric primitive is a very small (but crucial) cog in a very big machine,

Some Constants

- A symmetric primitive is a very small (but crucial) cog in a very big machine,
- there are many different “big machines”, and

Some Constants

- A symmetric primitive is a very small (but crucial) cog in a very big machine,
- there are many different "big machines", and
- this has a huge influence on what the primitive looks like.

Plan of this Section

1 What is the Purpose of a Symmetric Primitive

2 “Advanced” Protocols

3 Symmetric Primitives for Advanced Protocols

4 Conclusion

Plan of this Section

1 What is the Purpose of a Symmetric Primitive

2 “Advanced” Protocols

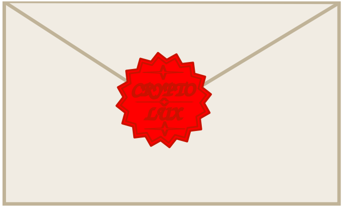
- General Introduction
- Different Protocols for Different Goals
- One Approach to Rule Them All (?): Arithmetization

3 Symmetric Primitives for Advanced Protocols

4 Conclusion

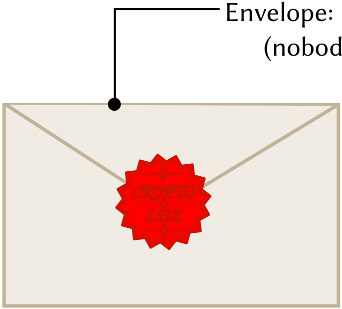
Securing Data

Usually, we secure data (at rest or in transit).



Securing Data

Usually, we secure data (at rest or in transit).

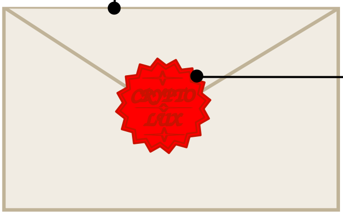


Envelope: Confidentiality
(nobody can read it)

Securing Data

Usually, we secure data (at rest or in transit).

Envelope: Confidentiality
(nobody can read it)

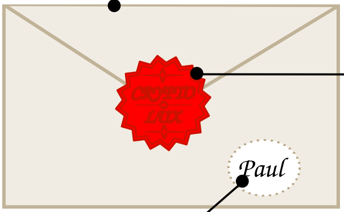


Seal: Integrity
(nobody can modify it)

Securing Data

Usually, we secure data (at rest or in transit).

Envelope: Confidentiality
(nobody can read it)



Seal: Integrity
(nobody can modify it)

Signature: Authentication
(it was written by the right person)

Securing Computation

More and more protocols intend to secure computations.

FHE	Fully Homomorphic Encryption
MPC	Multi Party Computations
ZK-*	Zero Knowledge- [proof, argument...]

Plan of this Section

1 What is the Purpose of a Symmetric Primitive

2 “Advanced” Protocols

- General Introduction
- Different Protocols for Different Goals
- One Approach to Rule Them All (?): Arithmetization

3 Symmetric Primitives for Advanced Protocols

4 Conclusion

FHE

Goal

Allow a third party to perform some operations on encrypted ciphertext that correspond to meaningful operations on the corresponding plaintext.

FHE

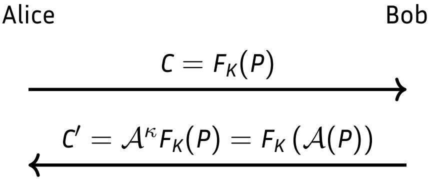
Goal

Allow a third party to perform some operations on encrypted ciphertext that correspond to meaningful operations on the corresponding plaintext. A form of commutation

FHE

Goal

Allow a third party to perform some operations on encrypted ciphertext that correspond to meaningful operations on the corresponding plaintext. A form of commutation



Alice

Bob

$C = F_K(P)$

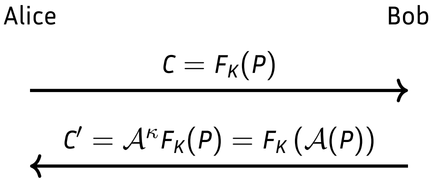
$C' = \mathcal{A}^{\kappa} F_K(P) = F_K\left(\mathcal{A}(P)\right)$

F_K is a homomorphic cipher,
not a block cipher!

FHE

Goal

Allow a third party to perform some operations on encrypted ciphertext that correspond to meaningful operations on the corresponding plaintext. A form of commutation



Alice

Bob

$C = F_K(P)$

$C' = \mathcal{A}^{\kappa} F_K(P) = F_K\left(\mathcal{A}(P)\right)$

F_K is a homomorphic cipher, **not** a block cipher!

An example of (not F)HE

XOR-ing a constant to a ciphertext obtained using a stream cipher XORs the same constant in the plaintext:

$$C \oplus t = (P \oplus K) \oplus t = (P \oplus t) \oplus K$$

Multi-Party Computations

Goal

Allow multiple parties to evaluate a function together even if some parties are not trustworthy.

Multi-Party Computations

Goal

Allow multiple parties to evaluate a function together even if some parties are not trustworthy.

Example

Shamir’s secret sharing

Multi-Party Computations

Goal

Allow multiple parties to evaluate a function together even if some parties are not trustworthy.

Example

Shamir’s secret sharing

Applications

- Masking (the side-channel attack counter-measure)
- MPC-in-the-head paradigm (e.g. for Picnic signatures)
- ...

Zero-Knowledge

Principle

I want to convince you I know a secret without revealing it

Zero-Knowledge

Principle

I want to convince you I know a secret without revealing it

A generic goal

To be able to prove/argue that a function was evaluated correctly without revealing its input.

Zero-Knowledge

Principle

I want to convince you I know a secret without revealing it

A generic goal

To be able to prove/argue that a function was evaluated correctly without revealing its input.

Ex: Sudoku time!

Zero-Knowledge

Principle

I want to convince you I know a secret without revealing it

A generic goal

To be able to prove/argue that a function was evaluated correctly without revealing its input.

Ex: Sudoku time!

Applications

- Offload computation to untrusted 3rd parties
- Electronic vote (mixnets)
- Signatures (e.g. with Solid)

Zero-Knowledge

Principle

I want to convince you I know a secret without revealing it

A generic goal

To be able to prove/argue that a function was evaluated correctly without revealing its input.

Ex: Sudoku time!

Applications

- Offload computation to untrusted 3rd parties
- Electronic vote (mixnets)
- Signatures (e.g. with Solid)
- BLOCKCHAIN!!1!

Plan of this Section

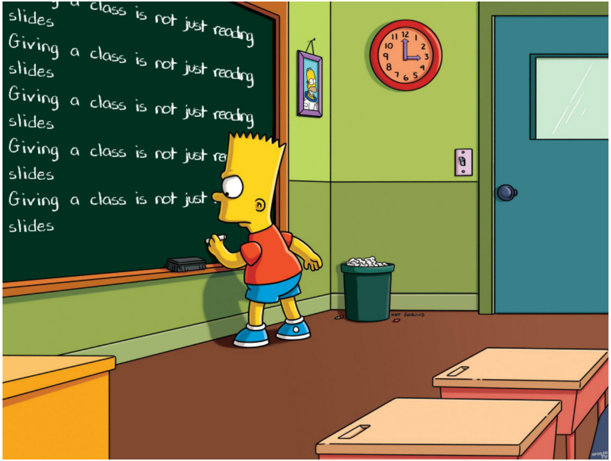
1 What is the Purpose of a Symmetric Primitive

2 “Advanced” Protocols
- General Introduction
- Different Protocols for Different Goals
- One Approach to Rule Them All (?): Arithmetization

3 Symmetric Primitives for Advanced Protocols

4 Conclusion

Arithmetization: General Principle


Giving a class is not just reading slides
Giving a class is not just reading slides
Giving a class is not just reading slides
Giving a class is not just rea
slides
Giving a class is not just
slides

A Basic Example of Arithmetization

“Arithmetization” depends on the subtleties of the protocol you work with!

A Basic Example of Arithmetization

"Arithmetization" depends on the subtleties of the protocol you work with!

Verifying if $y = c(ax + b)^{10} + x$ in R1CS

1. $t_0 = ax$
2. $t_1 = t_0 + b$
3. $t_2 = t_1 \times t_1$
4. $t_3 = t_2 \times t_2$
5. $t_4 = t_3 \times t_3$
6. $t_5 = t_2 \times t_4$
7. $t_6 = ct_5$
8. $y = t_6 + x$

A Basic Example of Arithmetization

“Arithmetization” depends on the subtleties of the protocol you work with!

Verifying if $y = c(ax + b)^{10} + x$ in R1CS

1. $t_0 = ax$
2. $t_1 = t_0 + b$
3. $t_2 = t_1 \times t_1$
4. $t_3 = t_2 \times t_2$
5. $t_4 = t_3 \times t_3$
6. $t_5 = t_2 \times t_4$
7. $t_6 = ct_5$
8. $y = t_6 + x$

A Basic Example of Arithmetization

"Arithmetization" depends on the subtleties of the protocol you work with!

Verifying if $y = c(ax + b)^{10} + x$ in R1CS

1. $t_0 = ax$
2. $t_1 = t_0 + b$
3. $t_2 = t_1 \times t_1$
4. $t_3 = t_2 \times t_2$
5. $t_4 = t_3 \times t_3$
6. $t_5 = t_2 \times t_4$
7. $t_6 = ct_5$
8. $y = t_6 + x$

This verification costs R1CS 4 constaints

A Basic Example of Arithmetization

"Arithmetization" depends on the subtleties of the protocol you work with!

Verifying if $y = c(ax + b)^{10} + x$ in R1CS

1. $t_0 = ax$
2. $t_1 = t_0 + b$
3. $t_2 = t_1 \times t_1$
4. $t_3 = t_2 \times t_2$
5. $t_4 = t_3 \times t_3$
6. $t_5 = t_2 \times t_4$
7. $t_6 = ct_5$
8. $y = t_6 + x$

This verification costs R1CS 4 constaints

- How to turn a computation into an arithmetic circuit depends on the operations allowed
- Its cost is also arithmetization-dependent—though low degree is usually welcome!

A not basic at all example of arithmetization

The cost of each operation depends on the arithmetization!

Plonk $\neq$ R1CS

A not basic at all example of arithmetization

The cost of each operation depends on the arithmetization!

Plonk $\neq$ R1CS

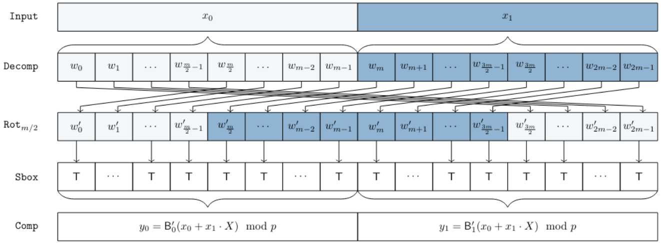
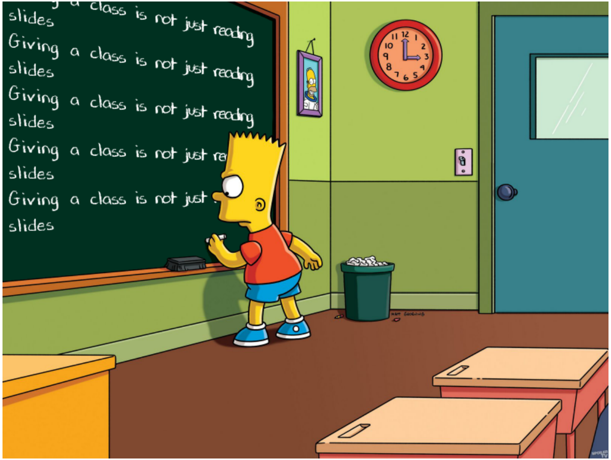


Figure 3: The Bar layer $\mathsf{B}' : \mathbb{F}_{p^n} \to \mathbb{F}_{p^n}$ for $n = 2$ in detail, including the decomposition, the rotation, the S-box, and the composition.

source: *Skyscraper: Fast Hashing on Big Primes*, `https://eprint.iacr.org/2025/058.pdf`

"Arithmetization-Oriented"? Evaluation vs. Verification

(the term was coined in [AAB$^+$20])


slides
Giving a class is not just reading slides
Giving a class is not just reading slides
Giving a class is not just re
slides
Giving a class is not just
slides

Symmetric Techniques for Advanced Protocols

MPC

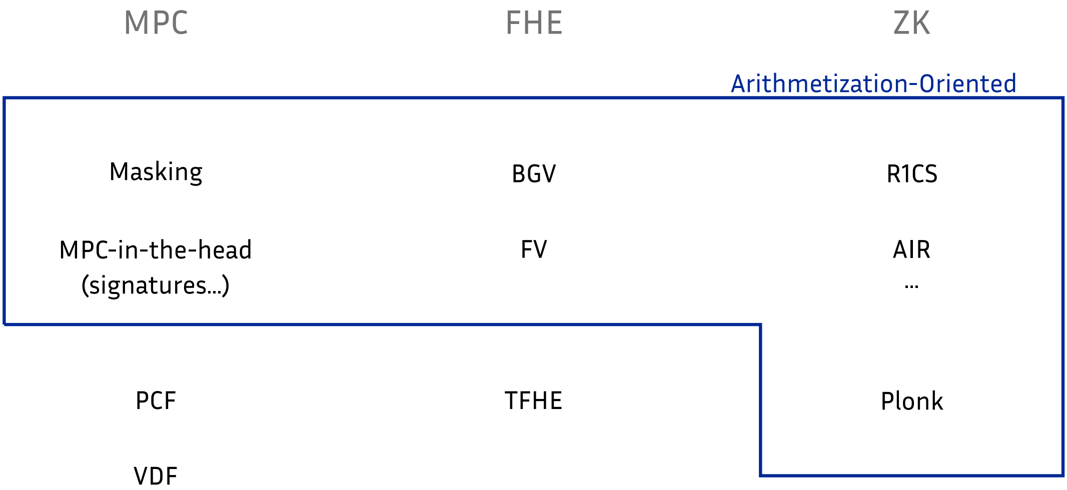
FHE

ZK

Symmetric Techniques for Advanced Protocols

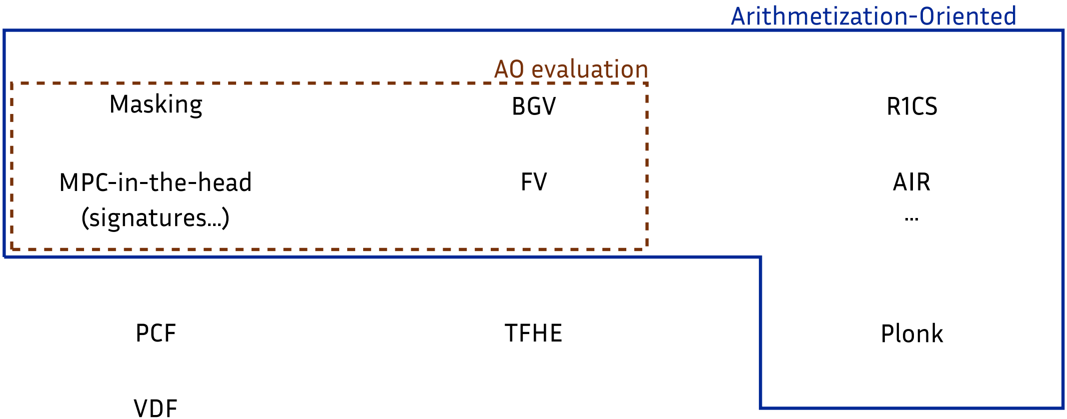
MPC	FHE	ZK
Masking	BGV	R1CS
MPC-in-the-head (signatures...)	FV	AIR ...
PCF	TFHE	Plonk
VDF		

Symmetric Techniques for Advanced Protocols



MPC	FHE	ZK
Masking	BGV	R1CS
MPC-in-the-head (signatures...)	FV	AIR ...
PCF	TFHE	Plonk
VDF		

Symmetric Techniques for Advanced Protocols



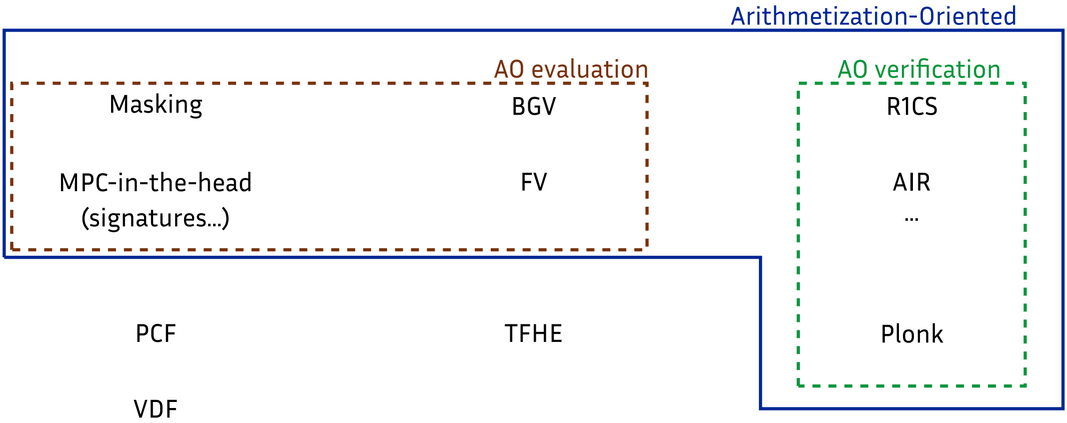
Arithmetization-Oriented

AO evaluation

MPC	FHE	ZK
Masking	BGV	R1CS
MPC-in-the-head (signatures...)	FV	AIR ...
PCF	TFHE	Plonk
VDF		

Symmetric Techniques for Advanced Protocols

MPC FHE ZK



Arithmetization-Oriented

AO evaluation

Masking

BGV

MPC-in-the-head (signatures...)

FV

AO verification

R1CS

AIR

...

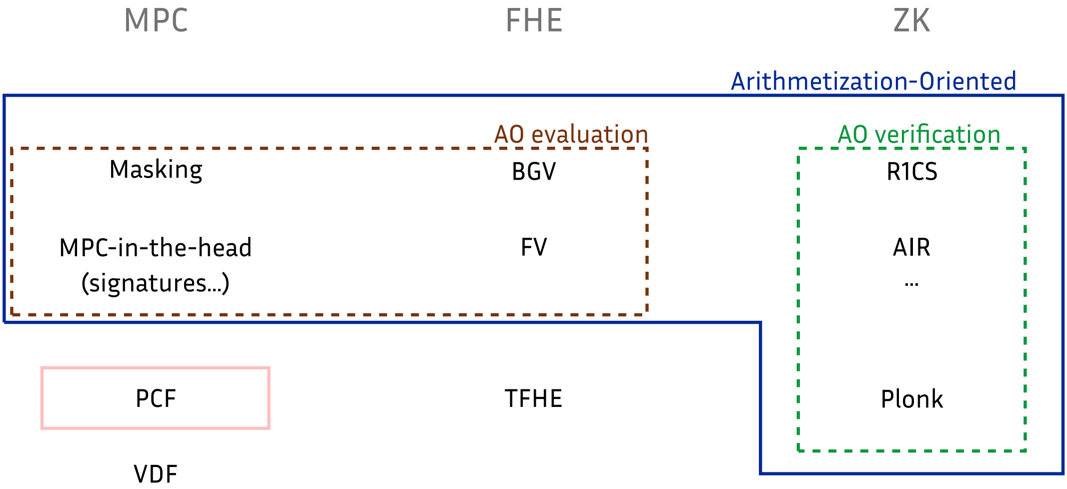
Plonk

PCF

TFHE

VDF

Symmetric Techniques for Advanced Protocols



MPC

FHE

ZK

Arithmetization-Oriented

AO evaluation

AO verification

Masking

BGV

R1CS

MPC-in-the-head (signatures...)

FV

AIR

...

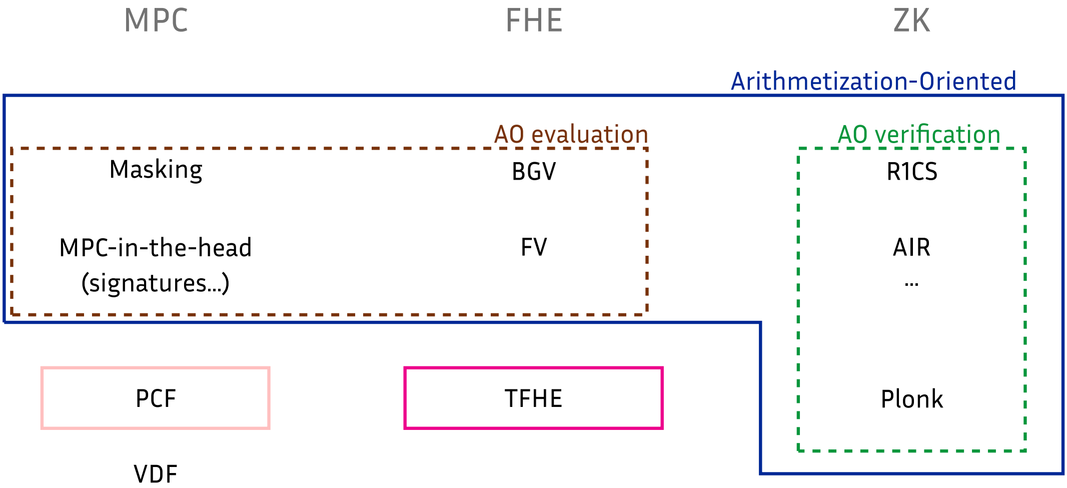
PCF

TFHE

Plonk

VDF

Symmetric Techniques for Advanced Protocols



MPC

FHE

ZK

Arithmetization-Oriented

AO evaluation

AO verification

Masking

BGV

R1CS

MPC-in-the-head
(signatures...)

FV

AIR
...

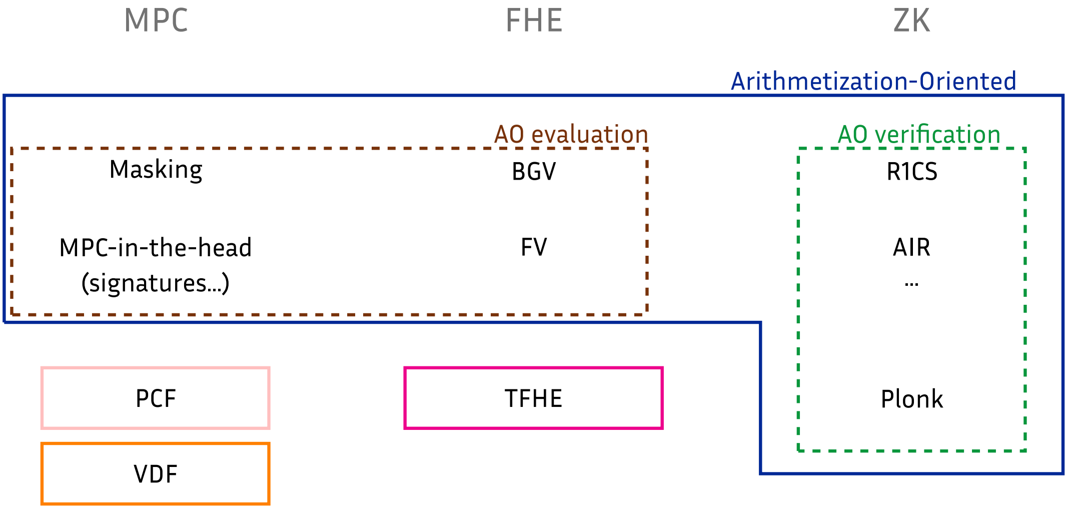
PCF

TFHE

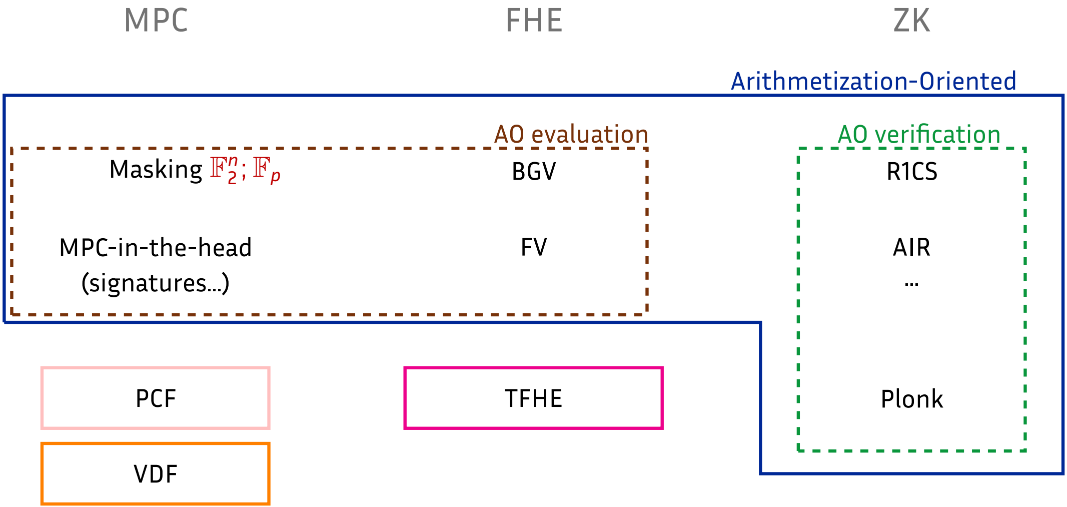
Plonk

VDF

Symmetric Techniques for Advanced Protocols



Symmetric Techniques for Advanced Protocols



MPC

FHE

ZK

Arithmetization-Oriented

AO evaluation

Masking $\mathbb{F}_2^n$; $\mathbb{F}_p$

BGV

MPC-in-the-head
(signatures...)

FV

AO verification

R1CS

AIR
...

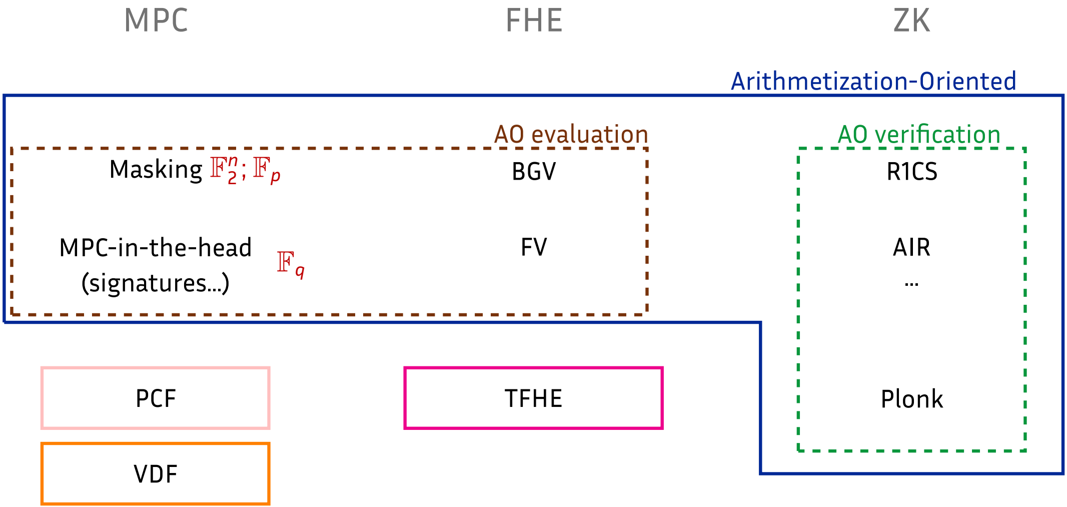
Plonk

PCF

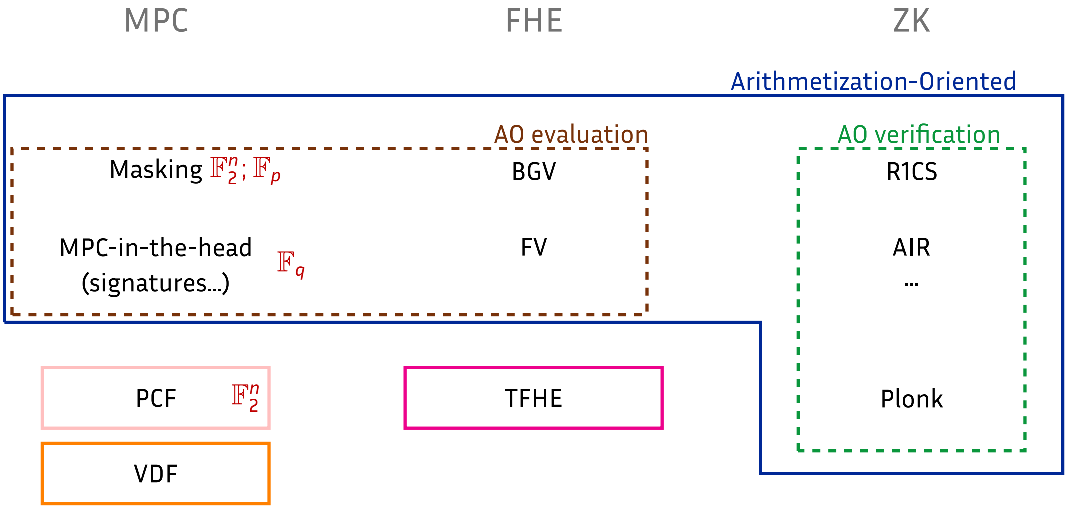
TFHE

VDF

Symmetric Techniques for Advanced Protocols



Symmetric Techniques for Advanced Protocols



MPC

FHE

ZK

Arithmetization-Oriented

AO evaluation

Masking $\mathbb{F}_2^n; \mathbb{F}_p$

BGV

MPC-in-the-head
(signatures...) $\mathbb{F}_q$

FV

AO verification

R1CS

AIR

...

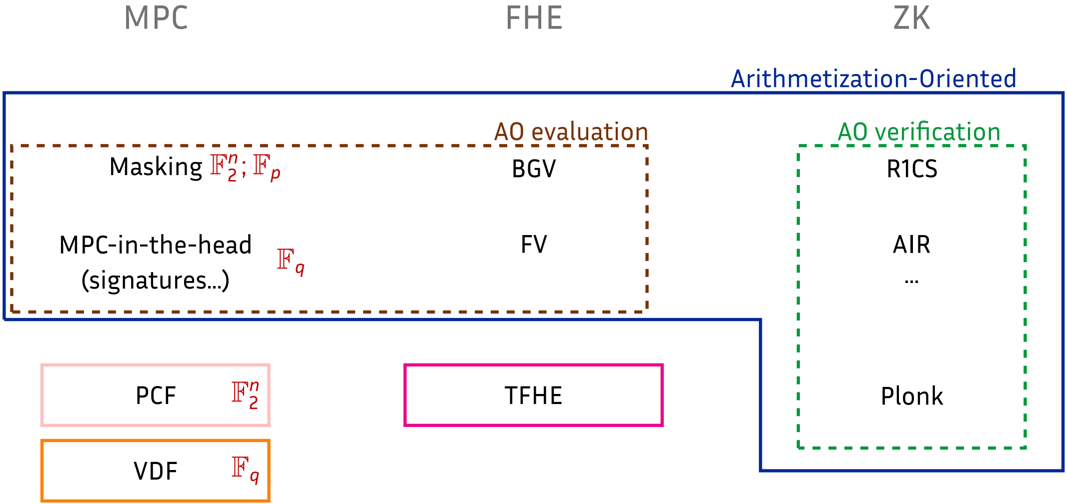
Plonk

PCF $\mathbb{F}_2^n$

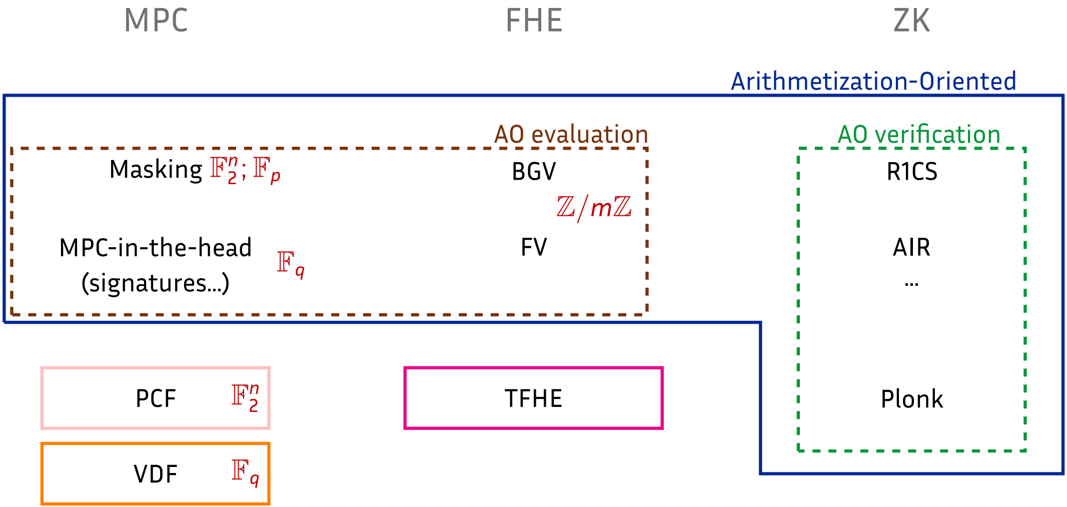
TFHE

VDF

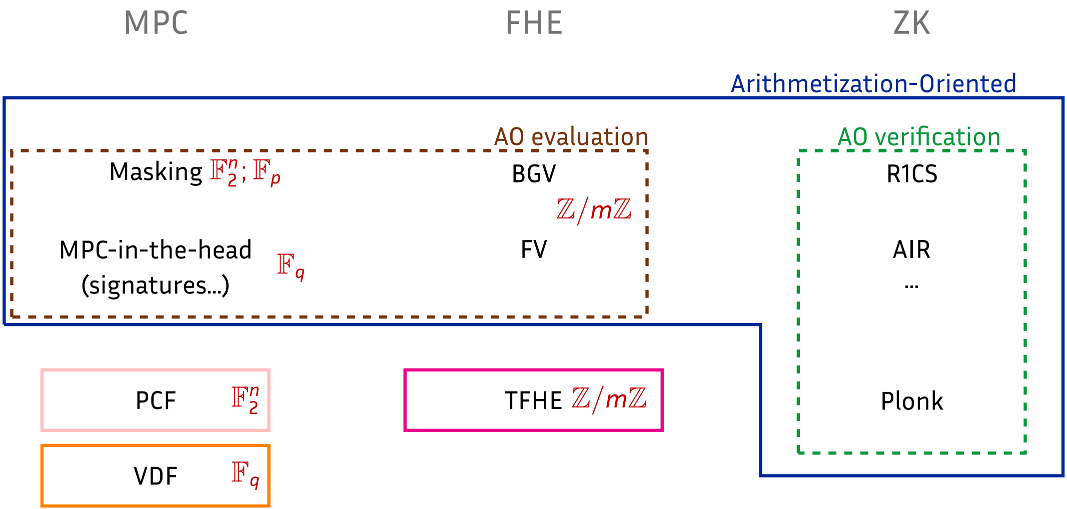
Symmetric Techniques for Advanced Protocols



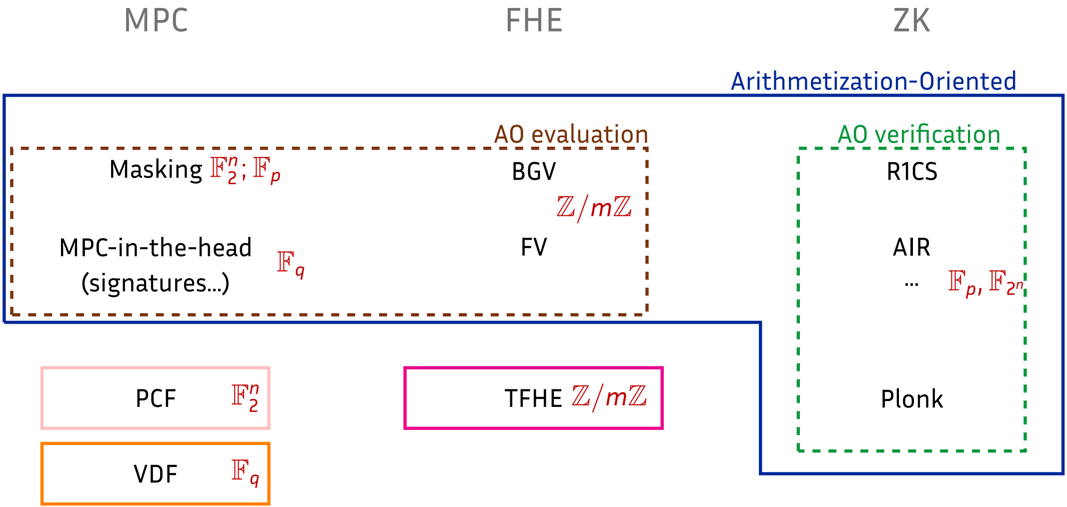
Symmetric Techniques for Advanced Protocols



Symmetric Techniques for Advanced Protocols



Symmetric Techniques for Advanced Protocols



MPC

FHE

ZK

Arithmetization-Oriented

AO evaluation

Masking $\mathbb{F}_2^n; \mathbb{F}_p$

MPC-in-the-head (signatures...) $\mathbb{F}_q$

BGV

$\mathbb{Z}/m\mathbb{Z}$

FV

AO verification

R1CS

AIR

... $\mathbb{F}_p, \mathbb{F}_{2^n}$

Plonk

PCF $\mathbb{F}_2^n$

VDF $\mathbb{F}_q$

TFHE $\mathbb{Z}/m\mathbb{Z}$

A Crucial Change?

$\mathbb{F}_q$ and $\mathbb{F}_2^n$ are not the same!

A Crucial Change?

$\mathbb{F}_q$ and $\mathbb{F}_2^n$ are not the same!

My Personal Opinion

- Indeed.

A Crucial Change?

$\mathbb{F}_q$ and $\mathbb{F}_2^n$ are not the same!

My Personal Opinion

- Indeed. In particular, q prime means no Frobenius automorphisms, meaning linear operations are only constant multiplications.

A Crucial Change?

$\mathbb{F}_q$ and $\mathbb{F}_2^n$ are not the same!

My Personal Opinion

- Indeed. In particular, q prime means no Frobenius automorphisms, meaning linear operations are only constant multiplications.
- Low degree arithmetization implies low degree algebraic modeling

beware of algebraic attacks!

A Crucial Change?

$\mathbb{F}_q$ and $\mathbb{F}_2^n$ are not the same!

My Personal Opinion

- Indeed. In particular, q prime means no Frobenius automorphisms, meaning linear operations are only constant multiplications.
- Low degree arithmetization implies low degree algebraic modeling

  beware of algebraic attacks!
- **However** good design approaches are *inherently* good design approaches

A Crucial Change?

$\mathbb{F}_q$ and $\mathbb{F}_2^n$ are not the same!

My Personal Opinion

- Indeed. In particular, q prime means no Frobenius automorphisms, meaning linear operations are only constant multiplications.
- Low degree arithmetization implies low degree algebraic modeling

  beware of algebraic attacks!
- **However** good design approaches are *inherently* good design approaches

Working over $\mathbb{F}_q$ (especially if low degree arithmetizations are needed) introduces new challenges, but solutions will rely on tried and true methods.

Plan of this Section

1 What is the Purpose of a Symmetric Primitive

2 “Advanced” Protocols

3 Symmetric Primitives for Advanced Protocols

4 Conclusion

Plan of this Section

1 What is the Purpose of a Symmetric Primitive

2 "Advanced" Protocols

3 Symmetric Primitives for Advanced Protocols

- FHE: Stream ciphers for transciphering
- MPC: low multiplicative depth, and PCF
- ZK: Hash function with AO verification

4 Conclusion

Transciphering

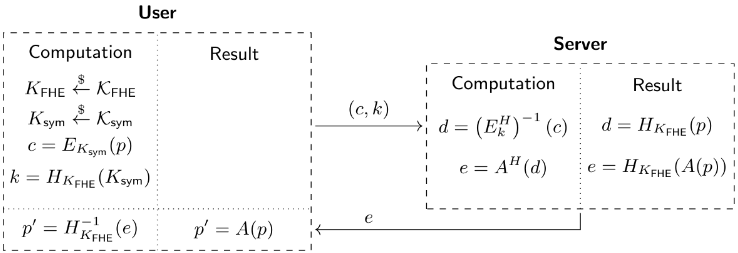

User
Computation
Result
$K_{\mathsf{FHE}} \xleftarrow{\$} \mathcal{K}_{\mathsf{FHE}}$
$K_{\mathsf{sym}} \xleftarrow{\$} \mathcal{K}_{\mathsf{sym}}$
$c = E_{K_{\mathsf{sym}}}(p)$
$k = H_{K_{\mathsf{FHE}}}(K_{\mathsf{sym}})$
$p' = H^{-1}_{K_{\mathsf{FHE}}}(e)$
$p' = A(p)$
(c, k)
e
Server
Computation
Result
$d = \left(E^H_k\right)^{-1}(c)$
$e = A^H(d)$
$d = H_{K_{\mathsf{FHE}}}(p)$
$e = H_{K_{\mathsf{FHE}}}(A(p))$

Fig. 1: The principle of transciphering, where E is a symmetric cipher (with secret key K_{sym} sampled from the space $\mathcal{K}_{\mathsf{sym}}$), H is a fully homomorphic cipher (with private key K_{FHE} sampled from the space $\mathcal{K}_{\mathsf{FHE}}$), E^H is a homomorphic evaluation of E, A corresponds to some arbitrary operations, and A^H to their homomorphic evaluation.

source: *Transistor: a TFHE-friendly Stream Cipher*

`https://eprint.iacr.org/2025/282`

The case of TFHE

Operates on $\mathbb{Z}/m\mathbb{Z}$, where m can be anything, though: more efficient if m is smaller.

Operations allowed

Linear Combinations $\sum_i \alpha_i x_i$, where the α_i are constant while x_i is input/key dependent.

- Costs almost nothing in terms of time/communication complexity...
- But noise increases

The case of TFHE

Operates on $\mathbb{Z}/m\mathbb{Z}$, where m can be anything, though: more efficient if m is smaller.

Operations allowed

Linear Combinations $\sum_i \alpha_i x_i$, where the α_i are constant while x_i is input/key dependent.

- Costs almost nothing in terms of time/communication complexity...
- But noise increases

PBS (Programmable BootStrap) $y \leftarrow S(x)$

- Very time consuming...
- But resets the noise to a base level

The case of TFHE

Operates on $\mathbb{Z}/m\mathbb{Z}$, where m can be anything, though: more efficient if m is smaller.

Operations allowed

Linear Combinations $\sum_i \alpha_i x_i$, where the α_i are constant while x_i is input/key dependent.

- Costs almost nothing in terms of time/communication complexity...
- But noise increases

PBS (Programmable BootStrap) $y \leftarrow S(x)$

- Very time consuming...
- But resets the noise to a base level
- Can be composed with arbitrary table lookups!

The case of TFHE

Operates on $\mathbb{Z}/m\mathbb{Z}$, where m can be anything, though: more efficient if m is smaller.

Operations allowed

Linear Combinations $\sum_i \alpha_i x_i$, where the α_i are constant while x_i is input/key dependent.

- Costs almost nothing in terms of time/communication complexity...
- But noise increases

PBS (Programmable BootStrap) $y \leftarrow S(x)$

- Very time consuming...
- But resets the noise to a base level
- Can be composed with arbitrary table lookups! $*$ *S-box sounds* $*$
- If the ring size is even, it is better if it is nega-cyclic ($S(x + 2^{n-1}) = -S(x)$)

TFHE: corresponding stream ciphers

Elisabeth-4 [CHMS22] ; $q = 2^4$

Uses a constant key register on which index-dependent non-linear functions are applied.

Can be linearized [GBJR23]

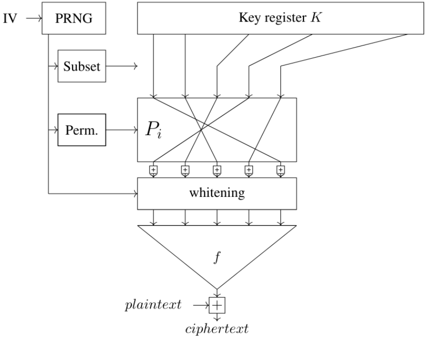


Fig. 1: The group filter permutator design

source: *Towards Case-Optimized Hybrid Homomorphic Encryption Featuring the Elisabeth Stream Cipher*

TFHE: corresponding stream ciphers

Elisabeth-4 [CHMS22] ; $q = 2^4$

Uses a constant key register on which index-dependent non-linear functions are applied.

Can be linearized [GBJR23]

Gabriel... [HMS23]

(Elisabeth-4 follow-ups)

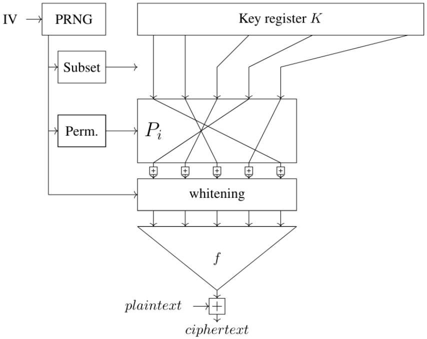


Fig. 1: The group filter permutator design

source: *Towards Case-Optimized Hybrid Homomorphic Encryption Featuring the Elisabeth Stream Cipher*

TFHE: corresponding stream ciphers

Elisabeth-4 [CHMS22] ; $q = 2^4$
Uses a constant key register on which index-dependent non-linear functions are applied.

Can be linearized [GBJR23]

Gabriel... [HMS23]
(Elisabeth-4 follow-ups)

FRAST [CCH$^+$24] ; $q = 2^4$ A block cipher in a CTR-mode variant.

See you at the rump session :D

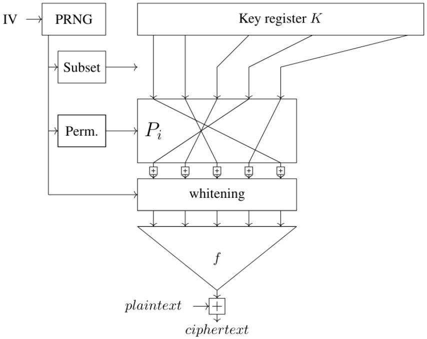


Fig. 1: The group filter permutator design

source: *Towards Case-Optimized Hybrid Homomorphic Encryption Featuring the Elisabeth Stream Cipher*

TFHE: corresponding stream ciphers

Elisabeth-4 [CHMS22] ; $q = 2^4$
Uses a constant key register on which index-dependent non-linear functions are applied.
Can be linearized [GBJR23]

Gabriel... [HMS23]
(Elisabeth-4 follow-ups)

FRAST [CCH$^+$24] ; $q = 2^4$ A block cipher in a CTR-mode variant.
See you at the rump session :D

Transistor [BBB$^+$25] ; $q = 2^4 + 1$
SNOW-like round structure
See you at Anne’s invited talk :D

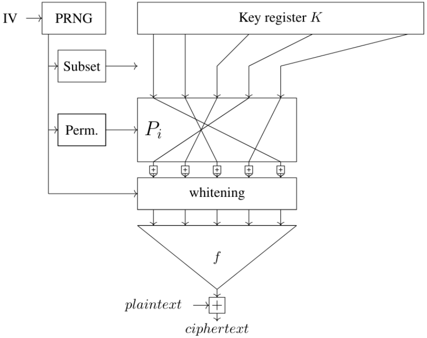


Fig. 1: The group filter permutator design

source: *Towards Case-Optimized Hybrid Homomorphic Encryption Featuring the Elisabeth Stream Cipher*

BGV/FV: corresponding stream ciphers

- ASTA $q = 2$ or large prime

  Use very few rounds with a low degree.

  Rely on large, randomly generated, nonce-dependent matrices.

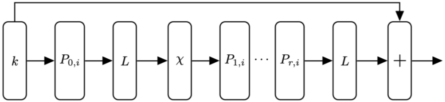


Figure 2: Generation of i-th block of DASTA.

source:

Dasta – Alternative Linear Layer for Rasta

BGV/FV: corresponding stream ciphers

·ASTA $q = 2$ or large prime
Use very few rounds with a low degree.
Rely on large, randomly generated, nonce-dependent matrices.

“Kreyvium” [CCF+16] $q = 2$
Basically Trivium!
Binary state updated with NLFSRs.

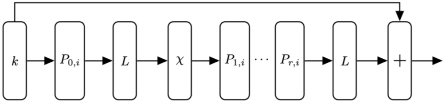


Figure 2: Generation of i-th block of Dasta.

source:
Dasta – Alternative Linear Layer for Rasta

BGV/FV: corresponding stream ciphers

·ASTA $q = 2$ or large prime
Use very few rounds with a low degree.
Rely on large, randomly generated, nonce-dependent matrices.

"Kreyvium" [CCF+16] $q = 2$
Basically Trivium!
Binary state updated with NLFSRs.

HERA [CHK+21] q large prime
A block cipher in a kind of CTR-mode variant.

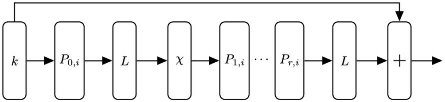


Figure 2: Generation of i-th block of DASTA.

source:
Dasta – Alternative Linear Layer for Rasta

Plan of this Section

1 What is the Purpose of a Symmetric Primitive

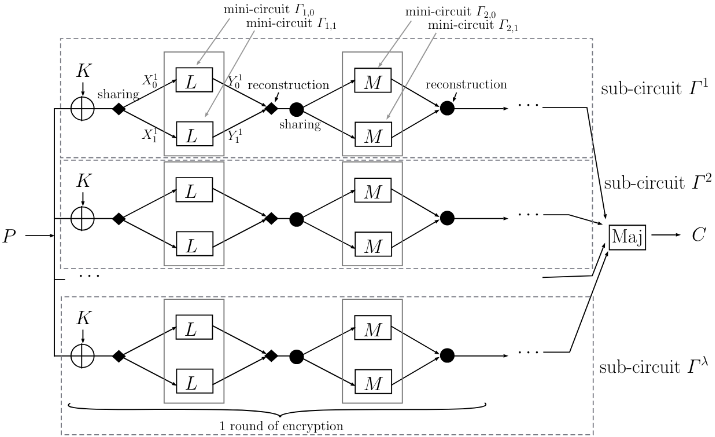
2 “Advanced” Protocols

3 Symmetric Primitives for Advanced Protocols

- FHE: Stream ciphers for transciphering
- MPC: low multiplicative depth, and PCF
- ZK: Hash function with AO verification

4 Conclusion

Trojan Resilience



source: *MOE: Multiplication Operated Encryption with Trojan Resilience*

`https://tosc.iacr.org/index.php/ToSC/article/view/8834`

MPC-Friendly Encryption

LowMC [ARS^{+}15] $q = 2$

SPN with partial layer of quadratic S-boxes.

Rely on large, randomly generated matrices.

Only one encryption/key; broken anyway

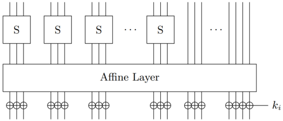


Fig. 1. Depiction of one round of encryption with LowMC.

source:

Ciphers for MPC and FHE

MPC-Friendly Encryption

LowMC [ARS$^+$15] $q = 2$

SPN with partial layer of quadratic S-boxes.

Rely on large, randomly generated matrices.

Only one encryption/key; broken anyway

Cimినion

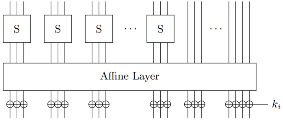


Fig. 1. Depiction of one round of encryption with LowMC.

source:

Ciphers for MPC and FHE

MPC-Friendly Encryption

LowMC [ARS+15] $q = 2$

SPN with partial layer of quadratic S-boxes.

Rely on large, randomly generated matrices.

Only one encryption/key; broken anyway

Cimінion [DGGK21] no specific constraints on q

3-branch Feistel network with a single multiplication/round.

small-pSquare [GMM+24] $p = q = 127$

Generalized Feistel network with low degree round function.

Optimized specifically for hardware masking.

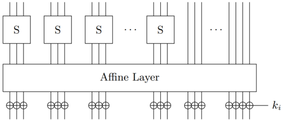


Fig. 1. Depiction of one round of encryption with LowMC.

source:

Ciphers for MPC and FHE

MPC-Friendly Encryption

LowMC [ARS$^+$15] $q = 2$

SPN with partial layer of quadratic S-boxes.

Rely on large, randomly generated matrices.

Only one encryption/key; broken anyway

Ciminion [DGGK21] no specific constraints on q

3-branch Feistel network with a single multiplication/round.

small-pSquare [GMM$^+$24] $p = q = 127$

Generalized Feistel network with low degree round function.

Optimized specifically for hardware masking.

MOE [BFL$^+$21] $q = 2^{128}$, $m = 2^{128}$

Dedicated structure with linear operations in $\mathbb{F}_q$ and $\mathbb{Z}/q\mathbb{Z}$. Intended for hardware trojan resilience.

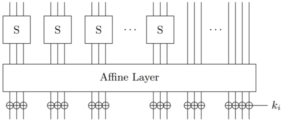


Fig. 1. Depiction of one round of encryption with LowMC.

source:

Ciphers for MPC and FHE

Pseudo-Correlated Functions

A new challenger!

At the start of some MPC protocols, it is necessary to share some bits that are correlated between the participants.

Very low multiplicative depth ·

Pseudo-Correlated Functions

A new challenger!

At the start of some MPC protocols, it is necessary to share some bits that are correlated between the participants.

Very low multiplicative depth · Very low data complexity ·

Pseudo-Correlated Functions

A new challenger!

At the start of some MPC protocols, it is necessary to share some bits that are correlated between the participants.

Very low multiplicative depth · Very low data complexity · Only known plaintext, no chosen plaintext!

Pseudo-Correlated Functions

A new challenger!

At the start of some MPC protocols, it is necessary to share some bits that are correlated between the participants.

Very low multiplicative depth · Very low data complexity · Only known plaintext, no chosen plaintext!

Crypto DarkMatter [BIP+18]

$$F_k(x) := \left(\sum_{i=0}^{n-1} k_i x_i \mod 2 + \sum_{i=0}^{n-1} k_i x_i \mod 3 \right) \mod 2, \quad \text{for } x \in \{0,1\}^n.$$

Pseudo-Correlated Functions

A new challenger!

At the start of some MPC protocols, it is necessary to share some bits that are correlated between the participants.

Very low multiplicative depth · Very low data complexity · Only known plaintext, no chosen plaintext!

Crypto DarkMatter [BIP+18]

$$F_k(x) := \left(\sum_{i=0}^{n-1} k_i x_i \mod 2 + \sum_{i=0}^{n-1} k_i x_i \mod 3 \right) \mod 2, \quad \text{for } x \in \{0,1\}^n.$$

VDLPN [BCG+20]

$$f_k(x) = \bigoplus_{i=1}^{D} \bigoplus_{j=1}^{w} \bigwedge_{\ell=1}^{i} (x_{i,j,\ell} \oplus k_{i,j,\ell}),$$

Plan of this Section

1 What is the Purpose of a Symmetric Primitive

2 “Advanced” Protocols

3 Symmetric Primitives for Advanced Protocols

- FHE: Stream ciphers for transciphering
- MPC: low multiplicative depth, and PCF
- ZK: Hash function with AO verification

4 Conclusion

ZK-Friendly Hash Functions

Poseidon [GLR+20] no specific constraints *q*

SPN with partial layer of low degree monomial S-boxes.

Full rounds – partial round – full rounds.

ZK-Friendly Hash Functions

Poseidon [GLR+20] no specific constraints q

SPN with partial layer of low degree monomial S-boxes.

Full rounds – partial round – full rounds.

Rescue [AAB+20] q large prime

SPN with low degree monomials and their inverses.

Most “AES-like”, also most secure at this stage.

ZK-Friendly Hash Functions

Poseidon [GLR+20] no specific constraints *q*

SPN with partial layer of low degree monomial S-boxes.

Full rounds – partial round – full rounds.

Rescue [AAB+20] *q* large prime

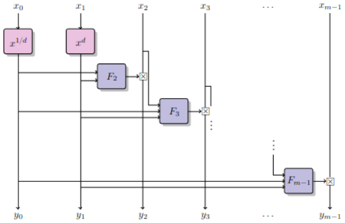
SPN with low degree monomials and their inverses.

Most “AES-like”, also most secure at this stage.

Griffin [GHR+23] *q* large prime

Feistel variant with multiplications instead.

Particularly vulnerable to algebraic attacks!



source: *Cryptanalysis and design of symmetric primitives defined over large finite fields*, PhD thesis of C. Bouvier

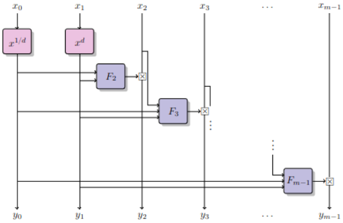
ZK-Friendly Hash Functions

Poseidon [GLR+20] no specific constraints q
SPN with partial layer of low degree monomial S-boxes.
Full rounds – partial round – full rounds.

Rescue [AAB+20] q large prime
SPN with low degree monomials and their inverses.
Most “AES-like”, also most secure at this stage.

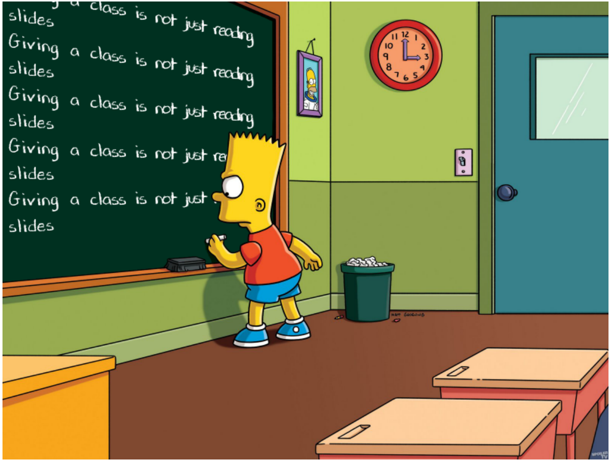
Griffin [GHR+23] q large prime
Feistel variant with multiplications instead.
Particularly vulnerable to algebraic attacks!

Anemoi [BBC+23]$q = 2^n$ or large prime
Uses the “Flystel”, a high degree S-box
CCZ-equivalent to a function of low degree.



source: *Cryptanalysis and design of symmetric primitives defined over large finite fields*, PhD thesis of C. Bouvier

Arithmetization-Oriented Verification: CCZ-equivalence?


a class is not just reading
slides
Giving a class is not just reading
slides
Giving a class is not just reading
slides
Giving a class is not just re
slides
Giving a class is not just
slides

Plan of this Section

1 What is the Purpose of a Symmetric Primitive

2 “Advanced” Protocols

3 Symmetric Primitives for Advanced Protocols

4 Conclusion

Conclusion

- The shape of symmetric primitives has always been dictated by external constraints

Conclusion

- The shape of symmetric primitives has always been dictated by external constraints
- New advanced protocols intended to secure computations need symmetric primitives

Conclusion

- The shape of symmetric primitives has always been dictated by external constraints
- New advanced protocols intended to secure computations need symmetric primitives
- The constraints they impose on the designs imply an unusual look at first glance

Conclusion

- The shape of symmetric primitives has always been dictated by external constraints
- New advanced protocols intended to secure computations need symmetric primitives
- The constraints they impose on the designs imply an unusual look at first glance
- The constraints they impose vary, and **we should be careful about lumping them together**

Conclusion

- The shape of symmetric primitives has always been dictated by external constraints
- New advanced protocols intended to secure computations need symmetric primitives
- The constraints they impose on the designs imply an unusual look at first glance
- The constraints they impose vary, and **we should be careful about lumping them together**
- A lot of new designs, very little cryptanalysis.

Conclusion

- The shape of symmetric primitives has always been dictated by external constraints
- New advanced protocols intended to secure computations need symmetric primitives
- The constraints they impose on the designs imply an unusual look at first glance
- The constraints they impose vary, and **we should be careful about lumping them together**
- A lot of new designs, very little cryptanalysis.

We need more cryptanalysis!

Conclusion

- The shape of symmetric primitives has always been dictated by external constraints
- New advanced protocols intended to secure computations need symmetric primitives
- The constraints they impose on the designs imply an unusual look at first glance
- The constraints they impose vary, and **we should be careful about lumping them together**
- A lot of new designs, very little cryptanalysis.

We need more cryptanalysis! **MORE!!**

Conclusion

- The shape of symmetric primitives has always been dictated by external constraints
- New advanced protocols intended to secure computations need symmetric primitives
- The constraints they impose on the designs imply an unusual look at first glance
- The constraints they impose vary, and **we should be careful about lumping them together**
- A lot of new designs, very little cryptanalysis.

We need more cryptanalysis! **MORE!!**

Coffee Break!

- Abdelrahaman Aly, Tomer Ashur, Eli Ben-Sasson, Siemen Dhooghe, and Alan Szepieniec.
  Design of symmetric-key primitives for advanced cryptographic protocols.
  IACR Trans. Symm. Cryptol., 2020(3):1–45, 2020.

- Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner.
  Ciphers for MPC and FHE.
  In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 430–454, Sofia, Bulgaria, April 26–30, 2015. Springer Berlin Heidelberg, Germany.

- Jules Baudrin, Sonia Belaïd, Nicolas Bon, Christina Boura, Anne Canteaut, Gaëtan Leurent, Pascal Paillier, Léo Perrin, Matthieu Rivain, Yann Rotella, and Samuel Tap.
  Transistor: a TFHE-friendly stream cipher.
  Cryptology ePrint Archive, Paper 2025/282, 2025.

- Clémence Bouvier, Pierre Briaud, Pyrros Chaidos, Léo Perrin, Robin Salen, Vesselin Velichkov, and Danny Willems.

New design techniques for efficient arithmetization-oriented hash functions: Anemoi permutations and Jive compression mode.
In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part III*, volume 14083 of *LNCS*, pages 507–539, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland.

Elette Boyle, Geoffroy Couteau, Niv Gilboa, Yuval Ishai, Lisa Kohl, and Peter Scholl.
Correlated pseudorandom functions from variable-density lpn.
In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1069–1080. IEEE, 2020.

Olivier Bronchain, Sebastian Faust, Virginie Lallemand, Gregor Leander, Léo Perrin, and François-Xavier Standaert.
MOE: Multiplication operated encryption with trojan resilience.
IACR Trans. Symm. Cryptol., 2021(1):78–129, 2021.

Dan Boneh, Yuval Ishai, Alain Passelègue, Amit Sahai, and David J. Wu.
Exploring crypto dark matter: New simple PRF candidates and their applications.
In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pages 699–729, Panaji, India, November 11–14, 2018. Springer, Cham, Switzerland.

- Anne Canteaut, Sergiu Carpov, Caroline Fontaine, Tancrède Lepoint, María Naya-Plasencia, Pascal Paillier, and Renaud Sirdey.
  Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression.
  In Thomas Peyrin, editor, *FSE 2016*, volume 9783 of *LNCS*, pages 313–333, Bochum, Germany, March 20–23, 2016. Springer Berlin Heidelberg, Germany.
- Mingyu Cho, Woohyuk Chung, Jincheol Ha, Jooyoung Lee, Eun-Gyeol Oh, and Mincheol Son.
  FRAST: TFHE-friendly cipher based on random S-boxes.
  IACR Transactions on Symmetric Cryptology, 2024(3):1–43, Sep. 2024.
- Jihoon Cho, Jincheol Ha, Seongkwang Kim, ByeongHak Lee, Joohee Lee, Jooyoung Lee, Dukjae Moon, and Hyojin Yoon.
  Transciphering framework for approximate homomorphic encryption.
  In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part III*, volume 13092 of *LNCS*, pages 640–669, Singapore, December 6–10, 2021. Springer, Cham, Switzerland.
- Orel Cosseron, Clément Hoffmann, Pierrick Méaux, and François-Xavier Standaert.
  Towards case-optimized hybrid homomorphic encryption - featuring the elisabeth stream cipher.

In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part III*, volume 13793 of *LNCS*, pages 32–67, Taipei, Taiwan, December 5–9, 2022. Springer, Cham, Switzerland.

Christoph Dobraunig, Lorenzo Grassi, Anna Guinet, and Daniël Kuijsters.
Ciminion: Symmetric encryption based on Toffoli-gates over large finite fields.
In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part II*, volume 12697 of *LNCS*, pages 3–34, Zagreb, Croatia, October 17–21, 2021. Springer, Cham, Switzerland.

Henri Gilbert, Rachelle Heim Boissier, Jérémy Jean, and Jean-René Reinhard.
Cryptanalysis of elisabeth-4.
In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part III*, volume 14440 of *LNCS*, pages 256–284, Guangzhou, China, December 4–8, 2023. Springer, Singapore, Singapore.

Lorenzo Grassi, Yonglin Hao, Christian Rechberger, Markus Schofnegger, Roman Walch, and Qingju Wang.
Horst meets fluid-SPN: Griffin for zero-knowledge applications.
In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part III*, volume 14083 of *LNCS*, pages 573–606, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland.

- Lorenzo Grassi, Reinhard Lüftenegger, Christian Rechberger, Dragos Rotaru, and Markus Schofnegger.
  On a generalization of substitution-permutation networks: The HADES design strategy.
  In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 674–704, Zagreb, Croatia, May 10–14, 2020. Springer, Cham, Switzerland.

- Lorenzo Grassi, Loïc Masure, Pierrick Méaux, Thorben Moos, and François-Xavier Standaert.
  Generalized feistel ciphers for efficient prime field masking - full version.
  Cryptology ePrint Archive, Report 2024/431, 2024.

- Clément Hoffmann, Pierrick Méaux, and François-Xavier Standaert.
  The patching landscape of elisabeth-4 and the mixed filter permutator paradigm.
  In Anupam Chattopadhyay, Shivam Bhasin, Stjepan Picek, and Chester Rebeiro, editors, *INDOCRYPT 2023, Part I*, volume 14459 of *LNCS*, pages 134–156, Goa, India, December 10–13, 2023. Springer, Cham, Switzerland.