PhD Defense

Development of Optimized Operations for Homomorphic Encryption

Nicolas Bon

14th November 2025

CryptoExperts - ENS-PSL

Manuscript: www.nicolasbon.com/phd

Table of Content

Part 1

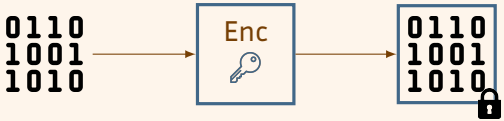
State of the Art

State of the Art

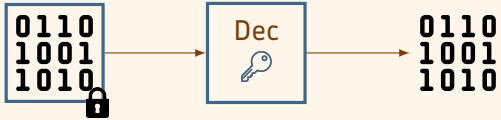
Fully Homomorphic Encryption

Cryptography

Encryption:



Decryption:



The three states of data

The three states of data

Data at rest

- Hard Drives
- Cloud Storage

The three states of data

Data at rest

- Hard Drives
- Cloud Storage

Data in transit

- Internet Traffic
- Messaging / E-mails

The three states of data

Data at rest

- Hard Drives
- Cloud Storage

Data in transit

- Internet Traffic
- Messaging / E-mails

The three states of data

Data at rest

- Hard Drives
- Cloud Storage

Data in transit

- Internet Traffic
- Messaging / E-mails

Data at work

- Server-side computations
- AI-based services

The three states of data

Data at rest

- Hard Drives
- Cloud Storage

Data in transit

- Internet Traffic
- Messaging / E-mails

Data at work

- Server-side computations
- AI-based services

Question

How to ensure secure computations ?

Fully Homomorphic Encryption

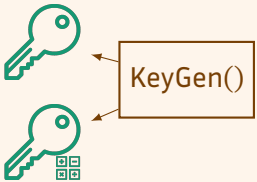
Client

0110
1001
1010

Server

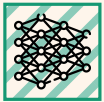
Fully Homomorphic Encryption

Client



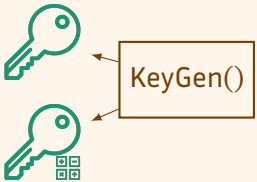
Server

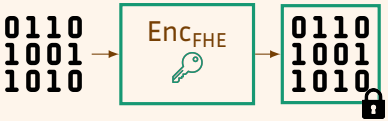
```
0110
1001
1010
```



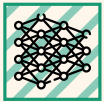
Fully Homomorphic Encryption

Client

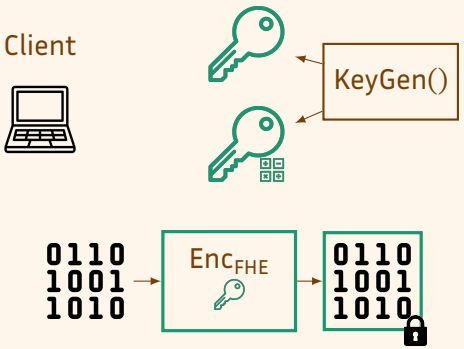


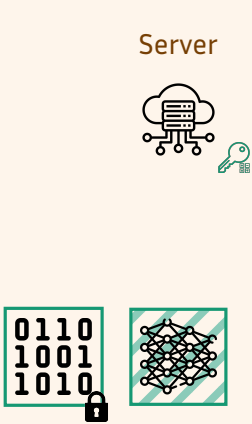


Server

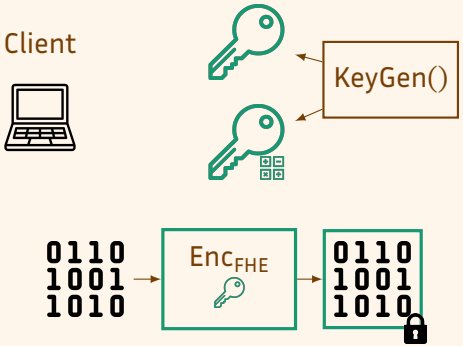


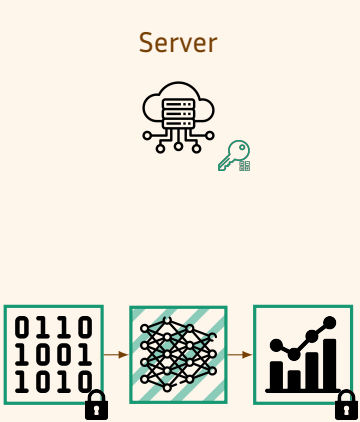
Fully Homomorphic Encryption



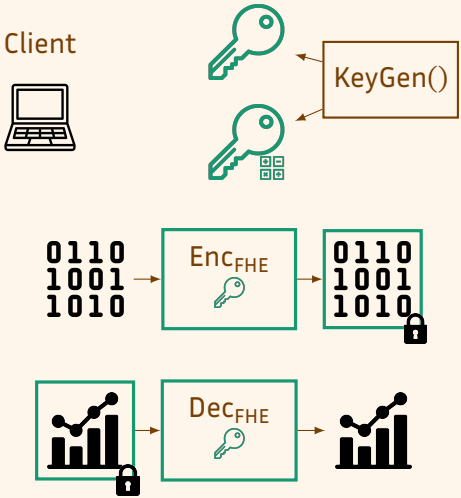


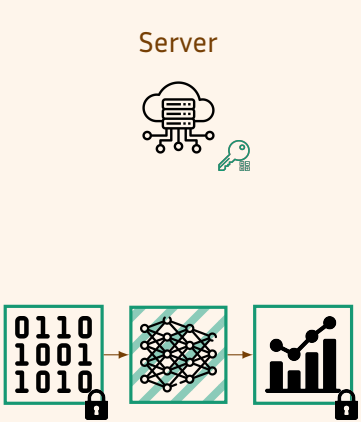
Fully Homomorphic Encryption



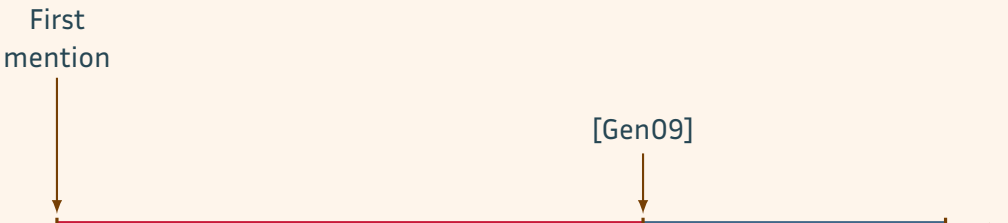


Fully Homomorphic Encryption





History of FHE

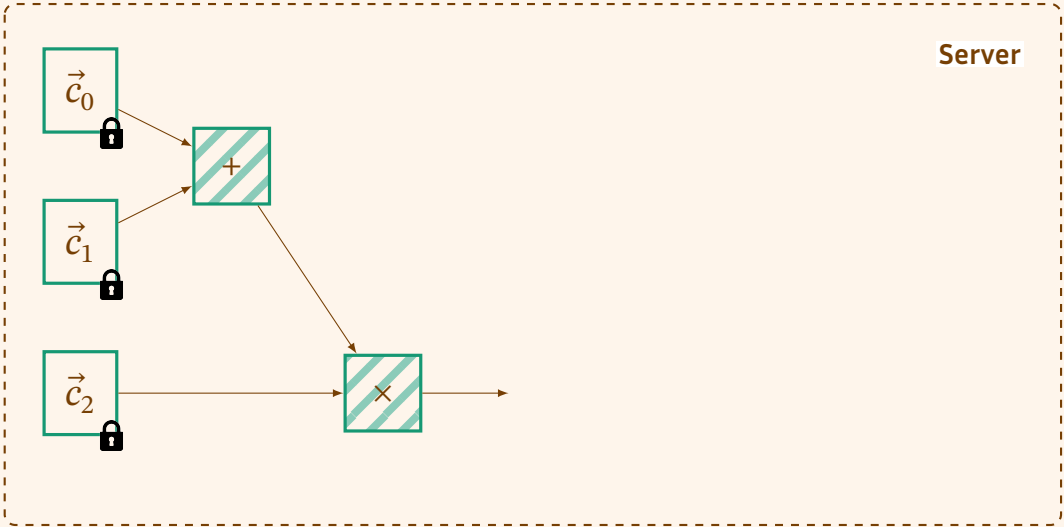


Partially Homomorphic:

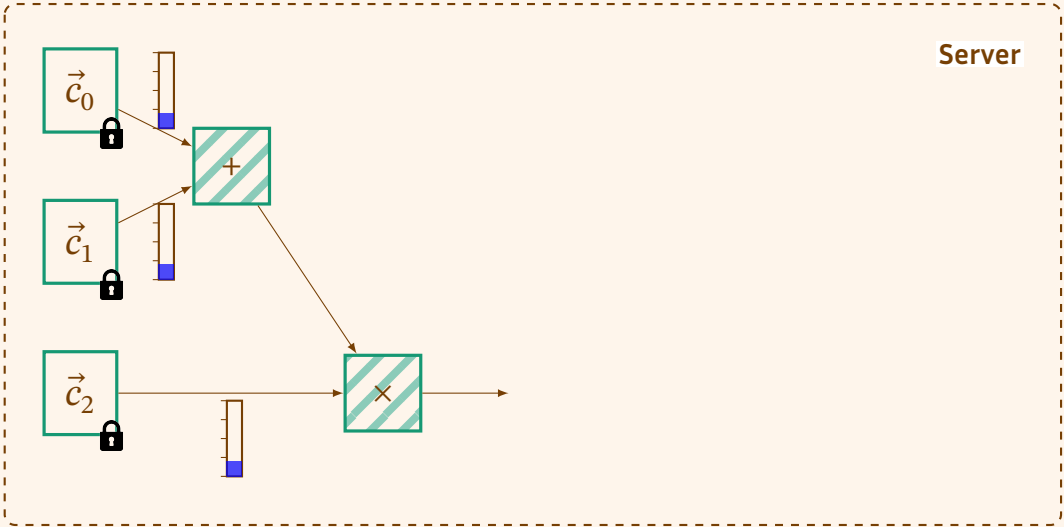
- Not every operation is supported
- Finite computational depth

Fully Homomorphic!

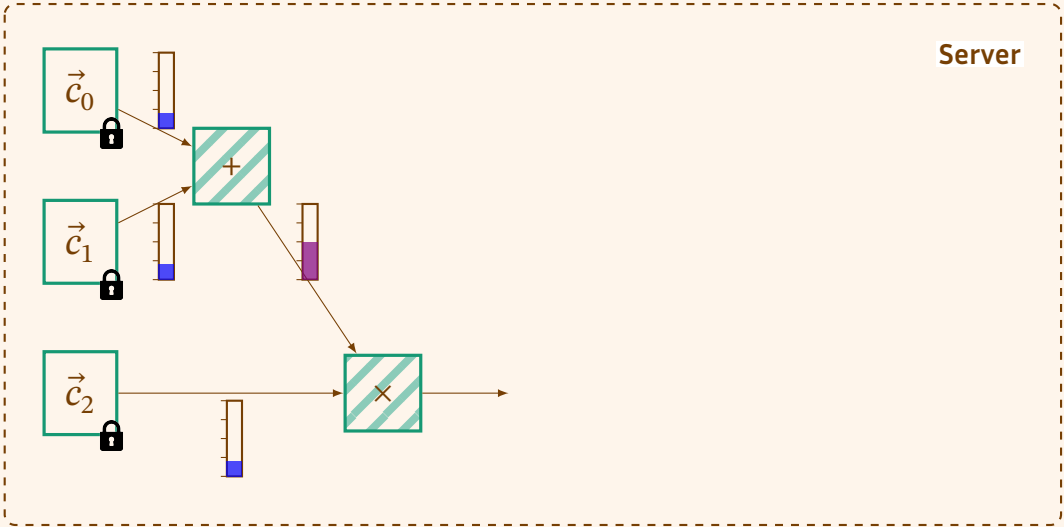
The Necessity of Bootstrapping



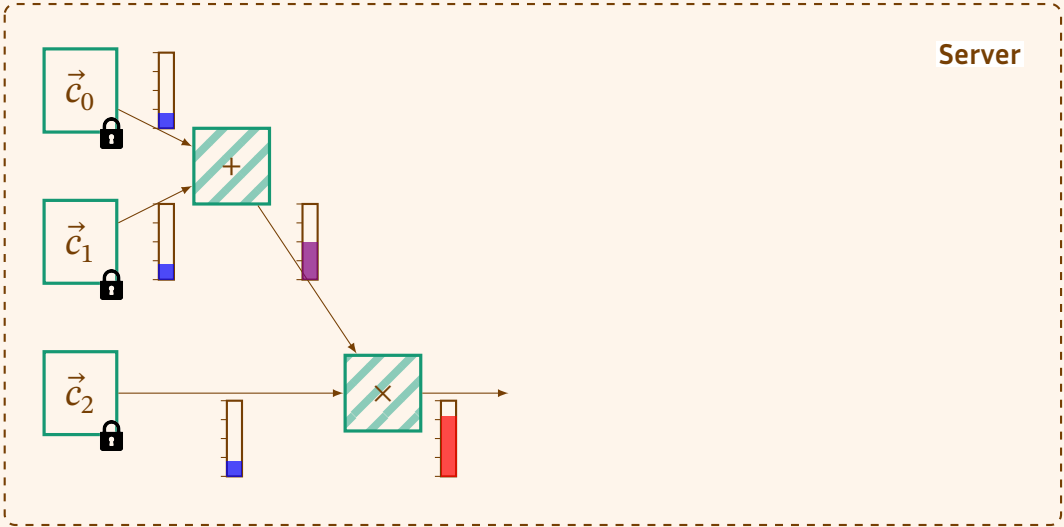
The Necessity of Bootstrapping



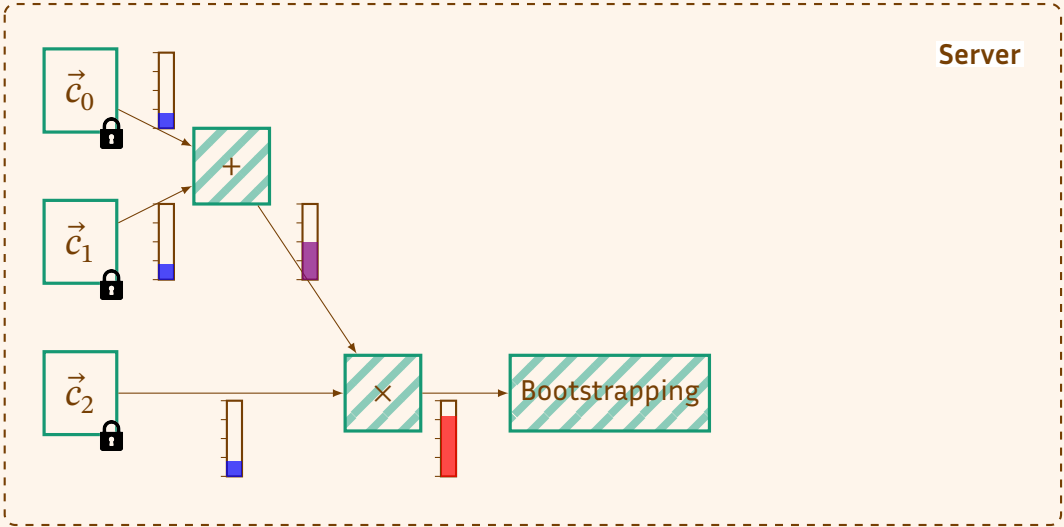
The Necessity of Bootstrapping



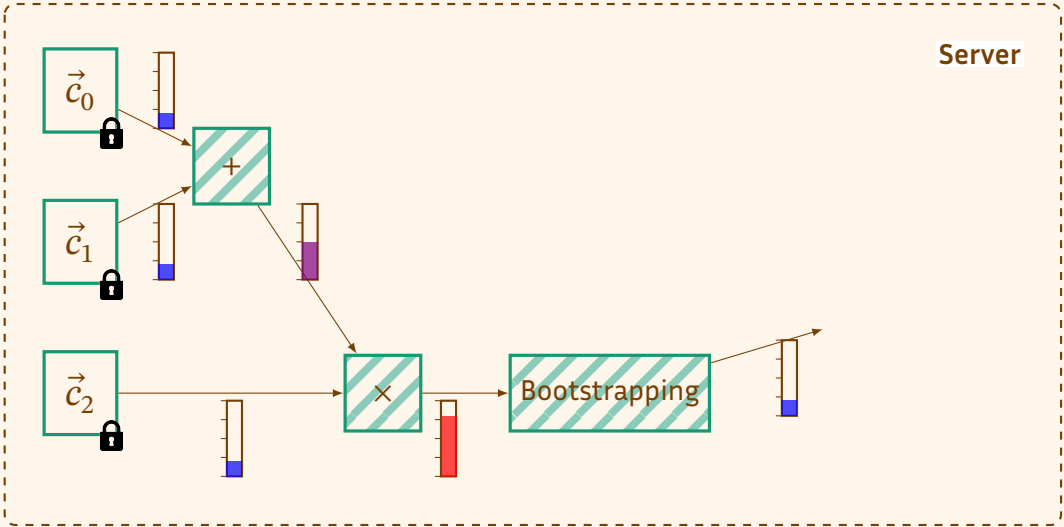
The Necessity of Bootstrapping



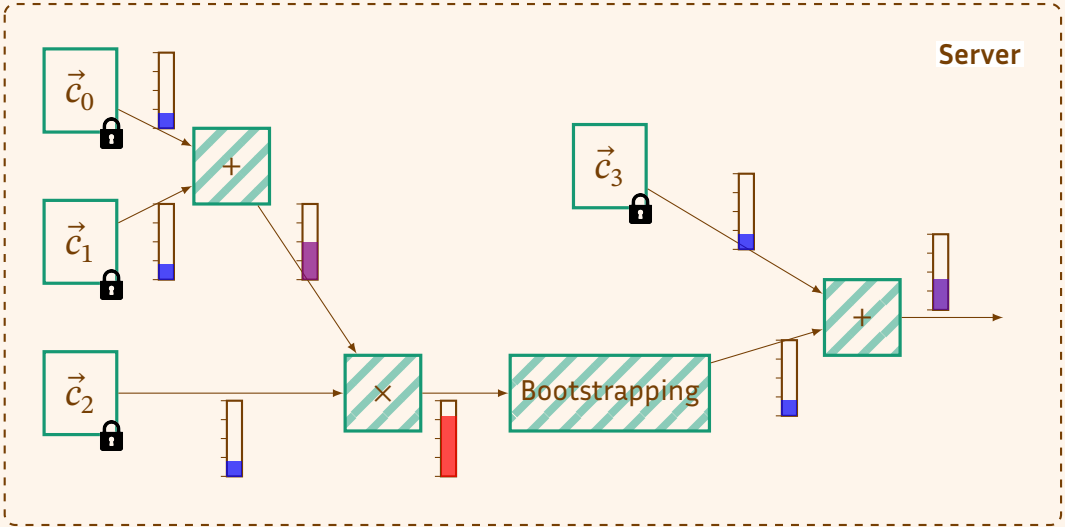
The Necessity of Bootstrapping



The Necessity of Bootstrapping

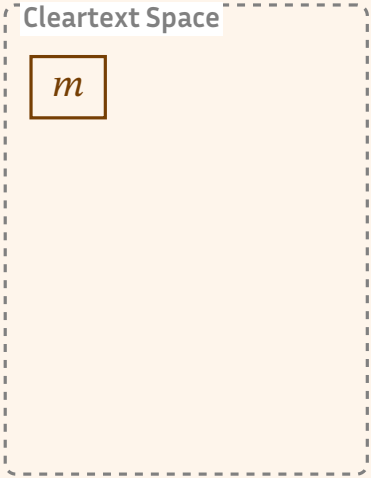


The Necessity of Bootstrapping

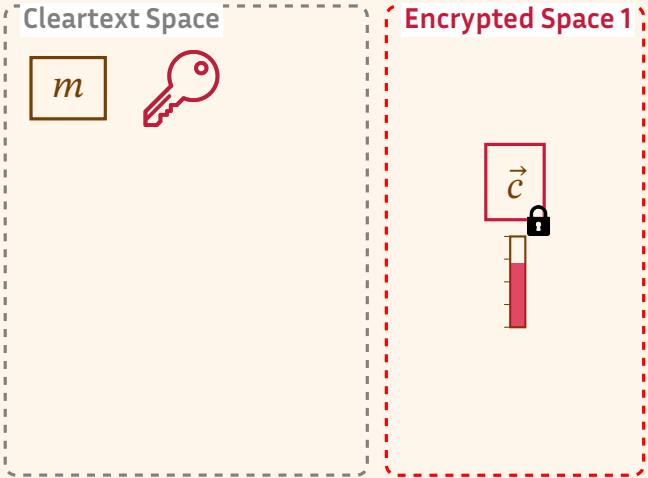


Gentry's blueprint for Bootstrapping

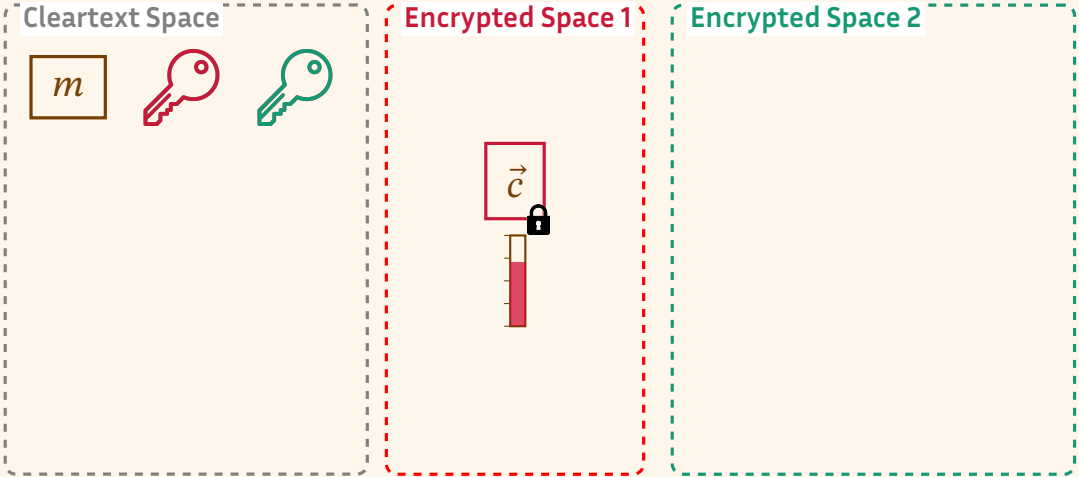
Gentry's blueprint for Bootstrapping



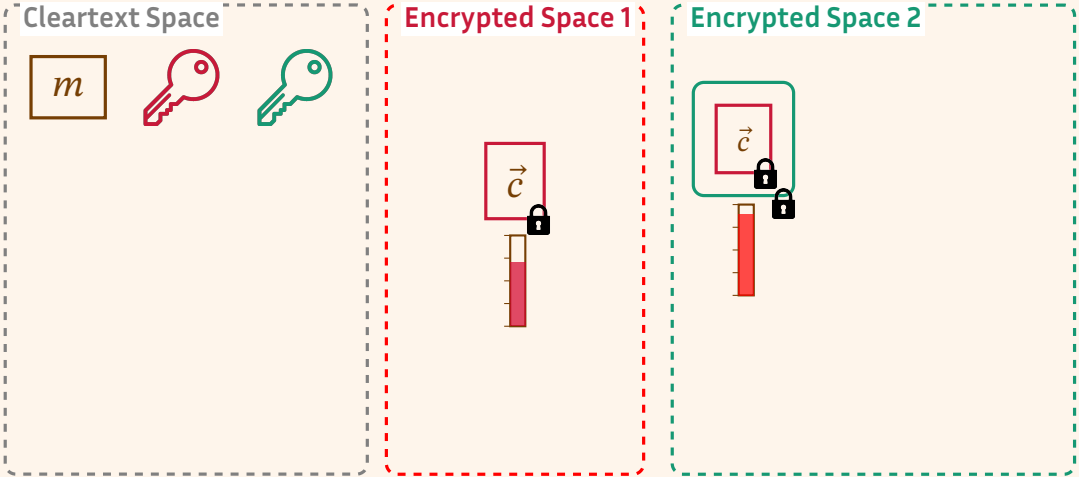
Gentry's blueprint for Bootstrapping



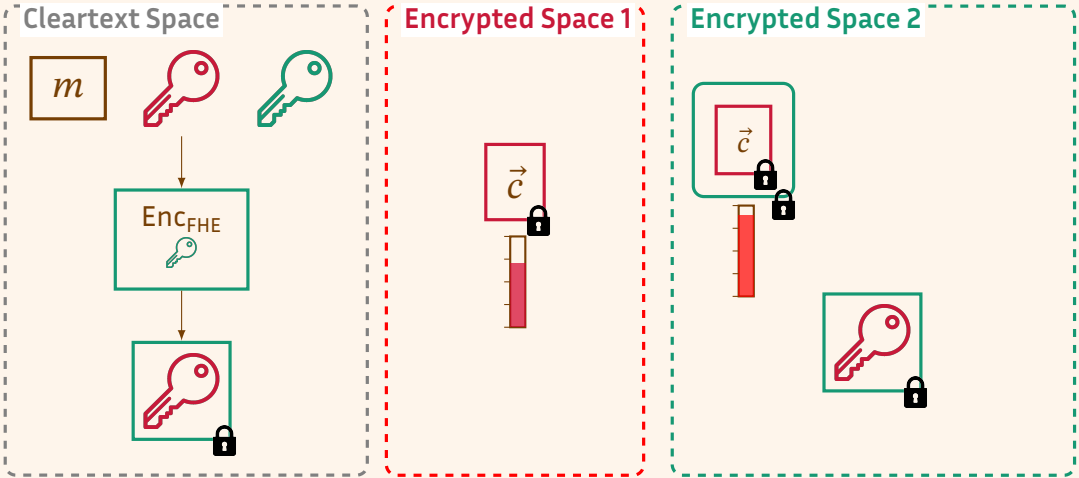
Gentry's blueprint for Bootstrapping



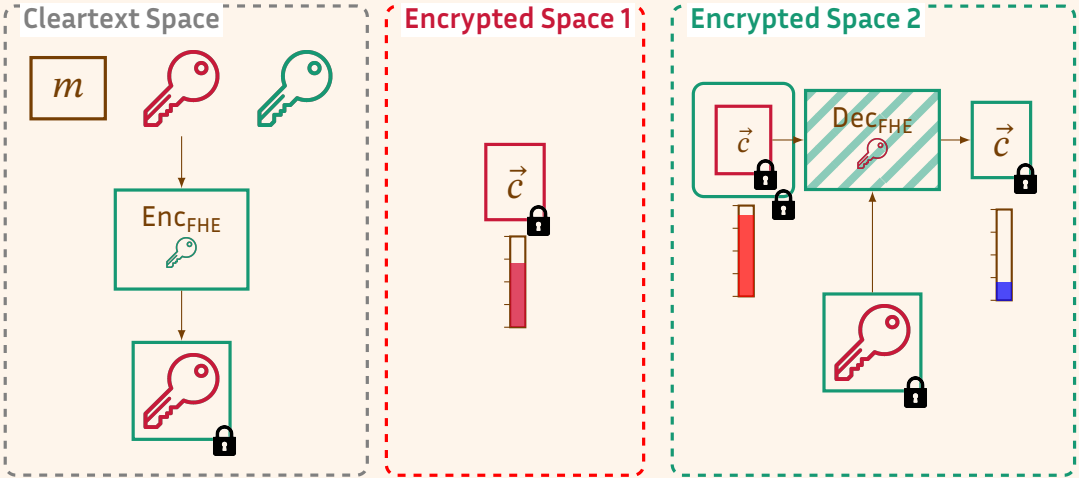
Gentry's blueprint for Bootstrapping



Gentry's blueprint for Bootstrapping



Gentry's blueprint for Bootstrapping

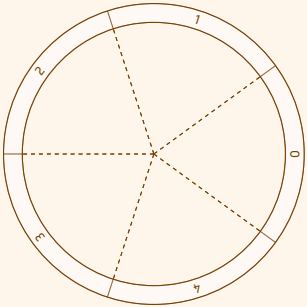


State of the Art

TFHE: FHE over the Torus

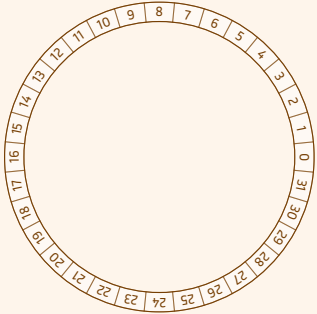
TFHE: Description of the scheme

Clear Space: $\mathbb{T}_p$



p has a size of a few bits.

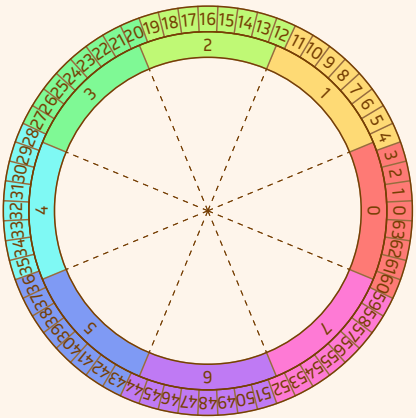
Encrypted Space: $\mathbb{T}_q$



$q = 2^{32}$ or 2^{64}.

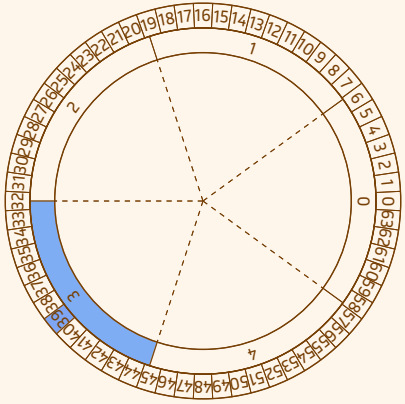
TFHE:Description of the scheme

Natural Embedding of $\mathbb{T}_p$ in $\mathbb{T}_q$



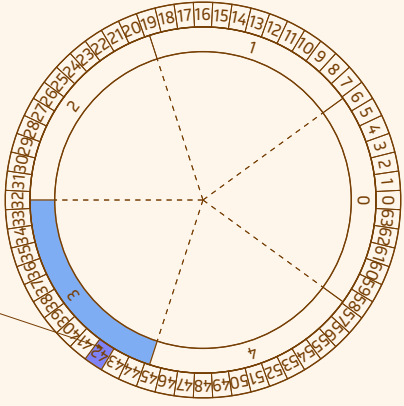
TFHE:Description of the scheme

Encoding of a message $m \in \mathbb{T}_p$



TFHE:Description of the scheme

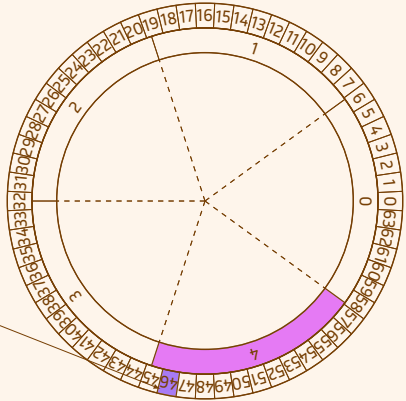
Encoding of a message $m \in \mathbb{T}_p$



Adds Random Gaussian Noise

TFHE:Description of the scheme

Encoding of a message $m \in \mathbb{T}_p$



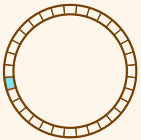
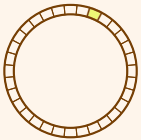
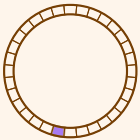
Too much noise causes decryption error!

Description of the scheme: Encryption Algorithm

Sampling of the secret key:

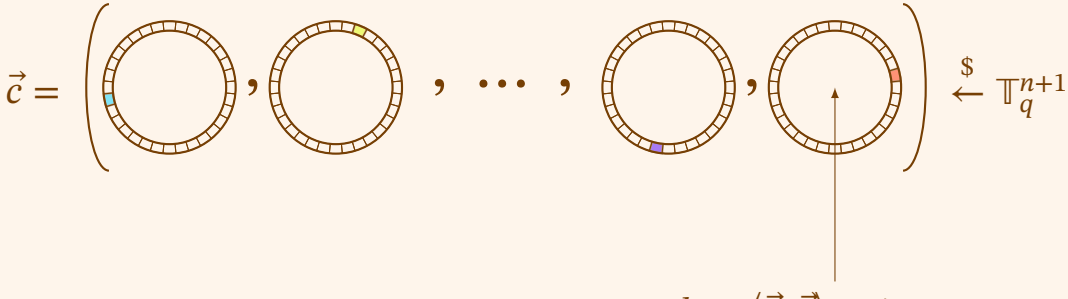
$$\vec{s} = (0, 1, \dots, 0) \xleftarrow{\$} \mathbb{B}^n$$

Sampling of a mask:

$$\vec{a} = \left(\text{  }, \text{  }, \cdots, \text{  } \right) \xleftarrow{\$} \mathbb{T}_q^n$$

Description of the scheme: Encryption Algorithm

Construction of ciphertext:



Description of the scheme: Encryption Algorithm

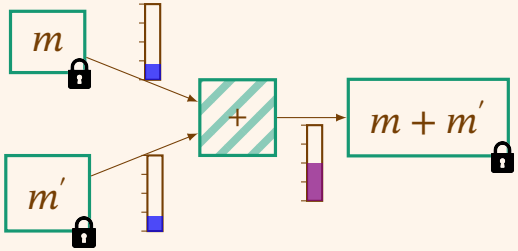
Construction of ciphertext:

Decryption:

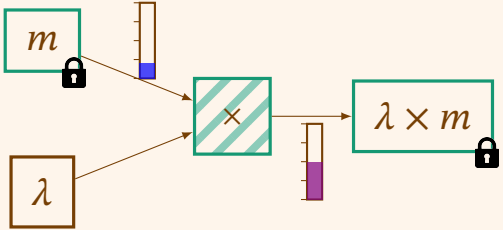
1. Recover the noisy message with $\Delta m + e = b - \langle \vec{a}, \vec{s} \rangle$.
2. Round to the closest plaintext: $m = \left\lfloor \frac{\Delta m + e}{\Delta} \right\rceil$.

Homomorphic Operations

Additions of ciphertexts:

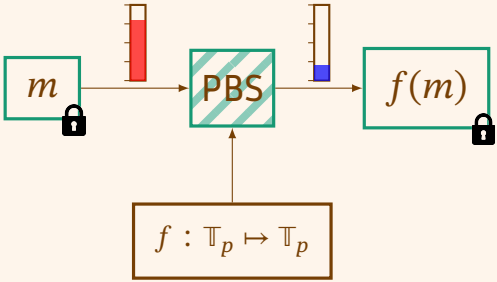


Multiplication of a ciphertext with a constant:



Programmable Bootstrapping

Main Feature: the **Programmable** Bootstrapping



Problem: PBS is very slow in large spaces

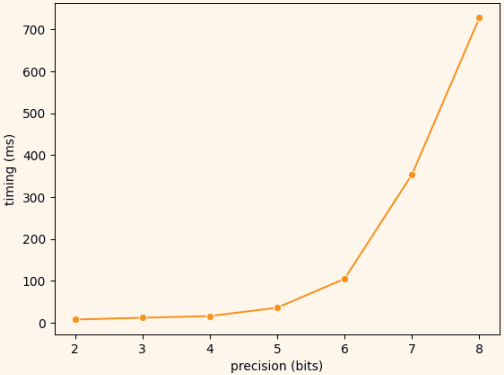


Figure: Degradation of the timing of a PBS, with respect to $\log_2(p)$.

State of the Art

Bootstrapping and Negacyclicity

Bootstrapping Outline

Decryption procedure is done is two steps:

- Computing $b - \langle \vec{a}, \vec{s} \rangle = m + e$
- Rounding to the closest plaintext: $\lfloor m + e \rceil = m$.

Bootstrapping Outline

Decryption procedure is done is two steps:

- Computing $b - \langle \vec{a}, \vec{s} \rangle = m + e$ **-> Easy**
- Rounding to the closest plaintext: $\lfloor m + e \rceil = m$.

Bootstrapping Outline

Decryption procedure is done is two steps:

- Computing $b - \langle \vec{a}, \vec{s} \rangle = m + e$ **-> Easy**
- Rounding to the closest plaintext: $\lfloor m + e \rceil = m.$ **-> More challenging**

Bootstrapping Outline

Decryption procedure is done is two steps:

- Computing $b - \langle \vec{a}, \vec{s} \rangle = m + e$ **-> Easy**
- Rounding to the closest plaintext: $\lfloor m + e \rceil = m$. **-> More challenging**

Question

How to perform rounding homomorphically?

Blind Rotation

Blind Rotation

Works in the polynomial ring $\mathbb{Z}[X]/(X^N + 1)$ (with N a power of two).

Blind Rotation

Works in the polynomial ring $\mathbb{Z}[X]/(X^N + 1)$ (with N a power of two).

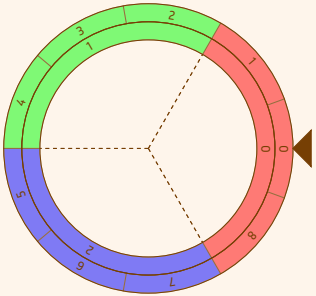
Let $v(X) = v_0 + v_1 \cdot X + \cdots + v_{N-1}X^{N-1}$ and $a \in \mathbb{Z}_{2N}$.

In this ring, something interesting happens:

$$X^{-a} \cdot v(X) = \begin{cases} v_a + v_{a+1} \cdot X + \ldots \text{ if } i \in [0, N[. \\ -v_a - v_{a+1} \cdot X + \ldots \text{ if } i \in [N, 2N[\end{cases}.$$

Blind Rotation

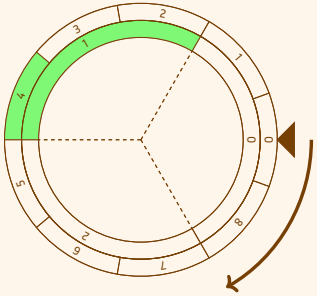
Blind rotation allows to homomorphically remove the noise:



$$\upsilon_0 + \upsilon_1 X + \upsilon_2 X^2 + \upsilon_3 X^3 + \upsilon_4 X^4 + \upsilon_5 X^5 + \upsilon_6 X^6 + \upsilon_7 X^7 + \upsilon_8 X^8$$

Blind Rotation

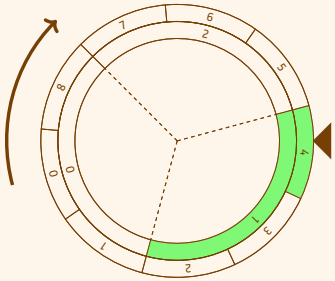
Blind rotation allows to homomorphically remove the noise:



$$X^{-4} \cdot \left(\upsilon_0 + \upsilon_1 X + \upsilon_2 X^2 + \upsilon_3 X^3 + \upsilon_4 X^4 + \upsilon_5 X^5 + \upsilon_6 X^6 + \upsilon_7 X^7 + \upsilon_8 X^8 \right)$$

Blind Rotation

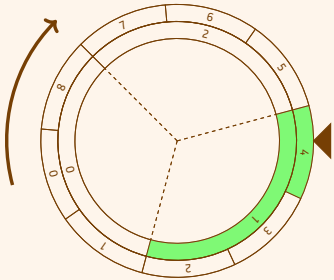
Blind rotation allows to homomorphically remove the noise:



$$\upsilon_4 + \upsilon_5 X + \upsilon_6 X^2 + \upsilon_7 X^3 + \upsilon_8 X^4 - \upsilon_0 X^5 - \upsilon_1 X^6 - \upsilon_2 X^7 - \upsilon_3 X^8$$

Blind Rotation

Blind rotation allows to homomorphically remove the noise:



$$\boxed{f(1)} + f(2)X + f(2)X^2 + f(2)X^3 - f(0)X^4 - f(0)X^5 - f(0)X^6 - f(1)X^7 - f(1)X^8$$

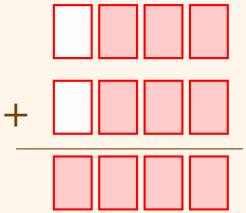
Blind Rotation

- Taking $q = 2N$ is unrealistic, so we modswitch the ciphertext from $\mathbb{Z}_q$ to $\mathbb{Z}_{2N}$.
- If the message is encoded lies between N and $2N$, then the output will be wrong.

Bit of Padding:

Pragmatic solution: enforce the MSB of ciphertexts to zero.

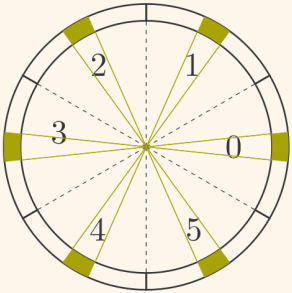
The bit-of-padding problem



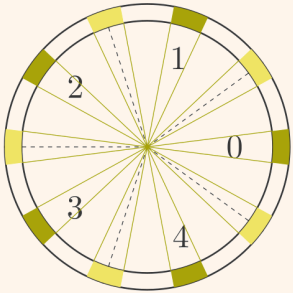
Problem

The bit-of-padding technique prevents from using the linear homomorphism!

Parity



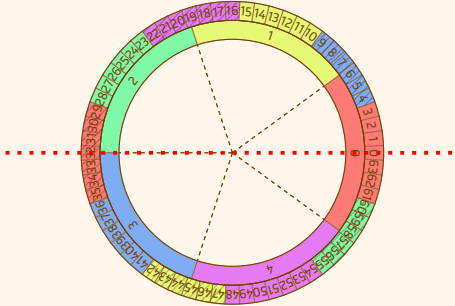
$p = 6$



$p = 5$

Parity

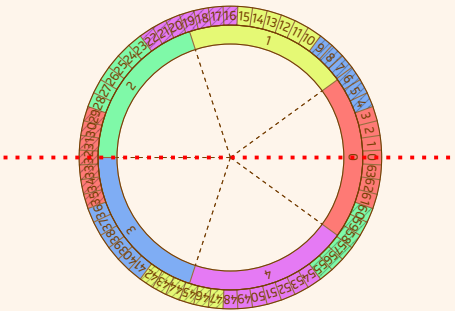
New Accumulator:



Odd-Modulus Accumulator:

Parity

New Accumulator:



Takeaway:

Odd moduli naturally solve the negacyclicity problem!

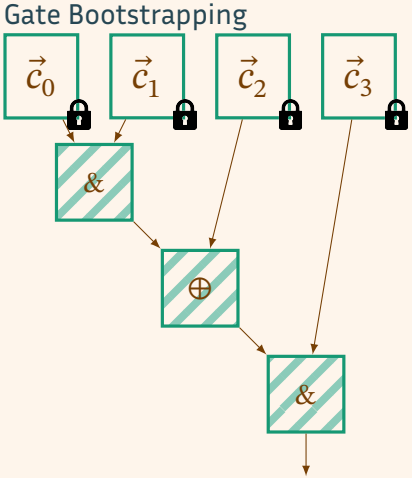
Part 2

Contributions

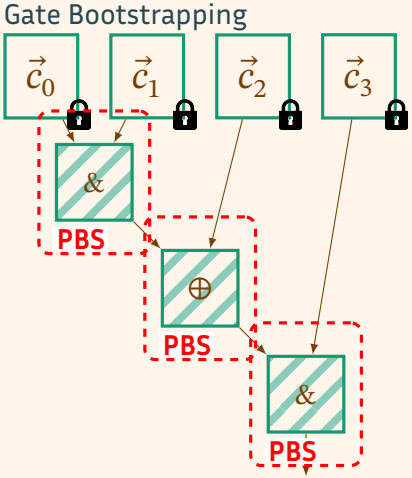
Contributions

Acceleration of Homomorphic Boolean Function

Gate Bootstrapping vs Look-Up Table

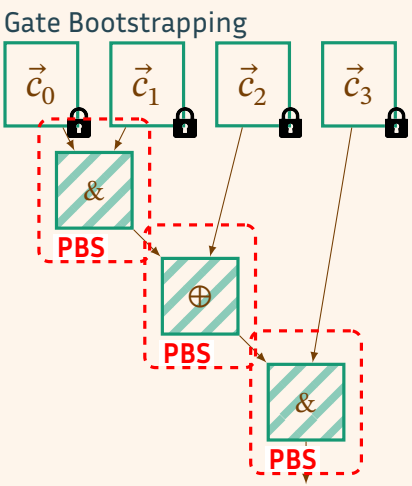


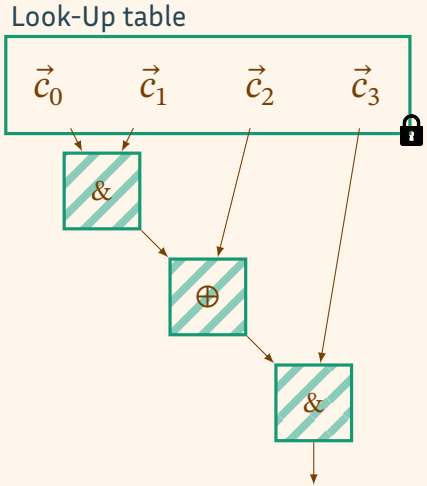
Gate Bootstrapping vs Look-Up Table



Problem: One Gate = One PBS.

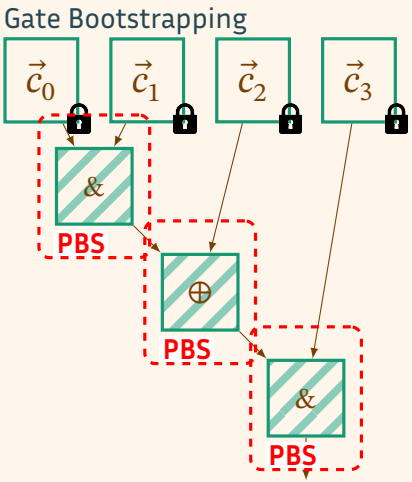
Gate Bootstrapping vs Look-Up Table



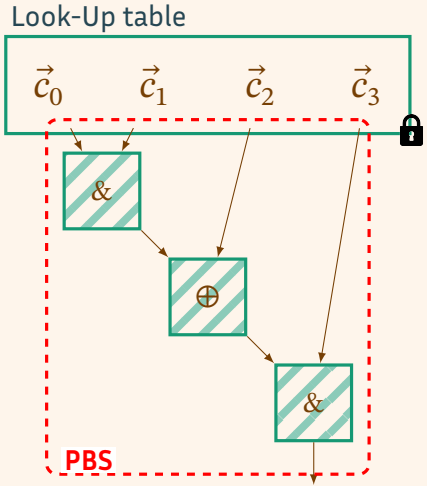


Problem: One Gate = One PBS.

Gate Bootstrapping vs Look-Up Table



Problem: One Gate = One PBS.

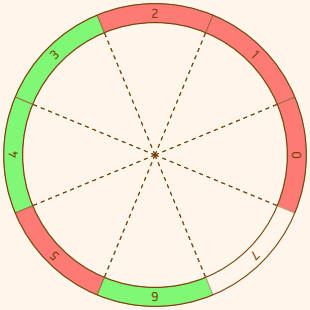


Problem: The PBS becomes very slow.

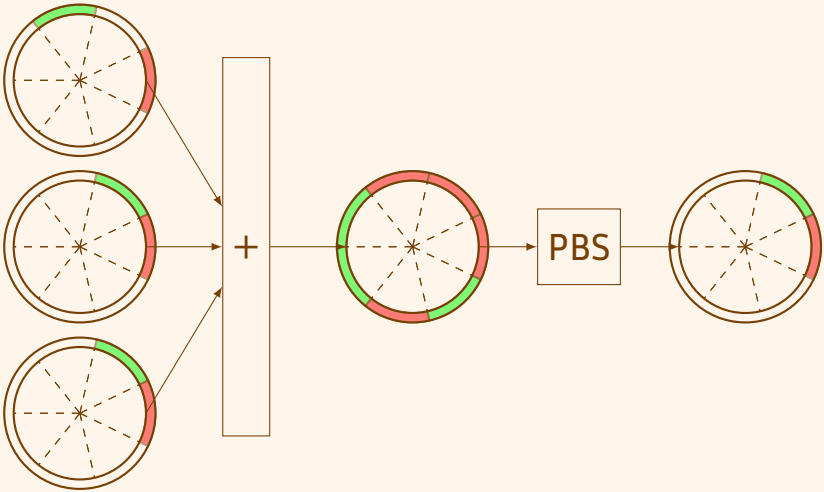
The p-encodings

A p-encoding is a function $\mathcal{E} : \mathbb{B} :\mapsto 2^{\mathbb{Z}_p}$.

$$\mathcal{E} : \begin{cases} 0 \mapsto \{\alpha_i\}_{0 \leq i \leq l_0} \\ 1 \mapsto \{\beta_i\}_{0 \leq i \leq l_1} \end{cases}$$



New Function Evaluation Algorithm



b_1	b_2	b_3	f
0	0	0	0
1	0	0	1
0	1	0	0
⋮	⋮	⋮	⋮

Advantages of the method

- One single bootstrapping to evaluate the whole function
- Plaintext space significantly smaller than 2^{ℓ}

Takeaway:

Better scaling than the traditional approaches

Boolean function on p-encodings

Question

For a given function $f : \mathbb{B}^\ell \mapsto \mathbb{B}$, how to find a valid set of p-encodings (and the best p)?

Boolean function on p-encodings

Question

For a given function $f : \mathbb{B}^\ell \mapsto \mathbb{B}$, how to find a valid set of p-encodings (and the best p)?

Exhaustive search is too costly when ℓ grows.

Reduction of the search space

If a solution exists, then it can be reduced to the form:

$$\mathcal{E}_i = \begin{cases} 0 \mapsto \{0\} \\ 1 \mapsto \{d_i\} \end{cases} \quad \text{with: } \mathcal{E}_0 = \begin{cases} 0 \mapsto \{0\} \\ 1 \mapsto \{1\} \end{cases}$$

Another point of view on the problem

Truth table of f:

b_1	b_2	b_3	$f(b_1, b_2, b_3)$
0	0	0	0
1	0	0	1
0	1	0	0
⋮	⋮	⋮	⋮

$$\begin{cases} 0 \cdot d_1 + 0 \cdot d_2 + 0 \cdot d_3 = r_0 \\ 1 \cdot d_1 + 0 \cdot d_2 + 0 \cdot d_3 = r_1 \\ 0 \cdot d_1 + 1 \cdot d_2 + 0 \cdot d_3 = r_2 \\ \vdots \end{cases}$$

Another point of view on the problem

Truth table of f:

b_1	b_2	b_3	$f(b_1, b_2, b_3)$
0	0	0	0
1	0	0	1
0	1	0	0
⋮	⋮	⋮	⋮

$$\begin{cases} 0 \cdot d_1 + 0 \cdot d_2 + 0 \cdot d_3 = r_0 \\ 1 \cdot d_1 + 0 \cdot d_2 + 0 \cdot d_3 = r_1 \\ 0 \cdot d_1 + 1 \cdot d_2 + 0 \cdot d_3 = r_2 \\ \vdots \end{cases}$$

Another point of view on the problem

By writing all the inequations $\neq$ we get:

$$\begin{cases} c_1^{(1)} d_1 + \cdots + c_\ell^{(1)} d_\ell \not\equiv 0 \pmod{p}, \\ c_1^{(2)} d_1 + \cdots + c_\ell^{(2)} d_\ell \not\equiv 0 \pmod{p}, \\ \vdots \\ c_1^{(2^\ell - 1)} d_1 + \cdots + c_\ell^{(2^\ell - 1)} d_\ell \not\equiv 0 \pmod{p}. \end{cases}$$

The search algorithm

- Our search algorithm finds a solution for a given p.
- To identify relevant values for p, we developed a **heuristic** method that finds an upper bound on the optimal p.

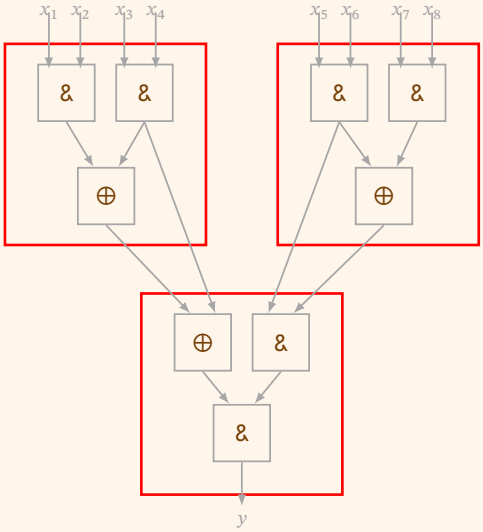
Experimental Results

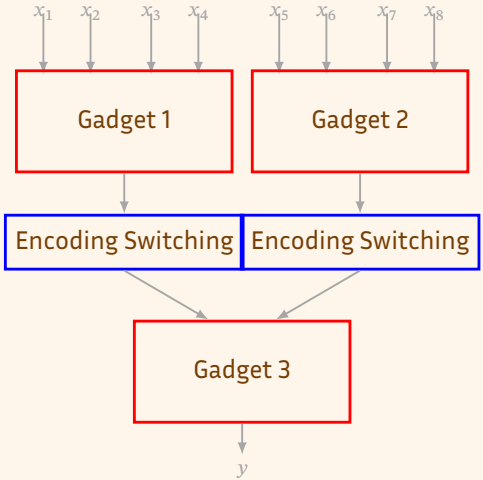
Primitive	Implementation	Performances
One full run of SIMON	Gate Bootstrapping	174 s
	[BSS+23]	128 s
	Our work	10 s
One warm-up phase of Trivium	Gate Bootstrapping	1498 s
	[BOS23] (estimation on our machine)	53 s
	Our work	32.8 s
One Full Keccak permutation	Gate Bootstrapping	30.7 min
	Our work	8.8 min
One Ascon hashing	Gate Bootstrapping	200s
	Our work	92 s

Experimental Results

Primitive	Implementation	Performances
One full run of SIMON $p = 9$	Gate Bootstrapping	174 s
	[BSS+23]	128 s
	Our work	10 s
One warm-up phase of Trivium $p = 9$	Gate Bootstrapping	1498 s
	[BOS23] (estimation on our machine)	53 s
	Our work	32.8 s
One Full Keccak permutation $p = 3$	Gate Bootstrapping	30.7 min
	Our work	8.8 min
One Ascon hashing $p = 17$	Gate Bootstrapping	200s
	Our work	92 s

Extension to larger circuits

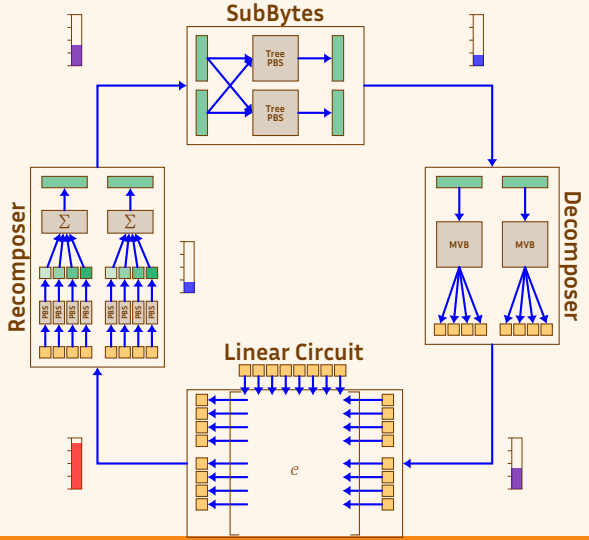




Performances of AES

One full evaluation of AES-128 ($\epsilon = 2^{-23}$) on one thread $p = 17$	[GHS12] †	18 min
	[CLT14] †	5 min
	[TCBS23]	270 s
	Our work	103 s
One full evaluation of AES-128 ($\epsilon = 2^{-40}$) on one thread $p = 17$	Gate Bootstrapping	234 s
	Our work (Real implementation)	135 s
	Our work (Theoretical timing with two keys)	105 s

Hippogryph: a better version of homomorphic AES



Contributions

Acceleration of large LUT Evaluation

Evaluation of large LUT

PBS is only efficient at small precision, so we cannot evaluate large LUT directly with it.

Question

How to decompose a LUT into small PBS operations?

An analogy with side-channels protection

- To protect the evaluation of LUT (e.g. S-box of AES) against side-channel attacks, masking is usually used.
- Masking AND gates is the most costly.
- Techniques to generate masked circuits minimizing the number of AND gates.

Question

Can we do the same, but minimizing the number of PBS calls?

Formalization of the problem

$$f : \mathbb{Z}_t \mapsto \mathbb{Z}_{t'}$$

Taking $t = s^n$, we manipulate blocks of size s. If s is not prime, we encode them in $\mathbb{F}_p$.

$$\mathbb{Z}_s \subseteq \mathbb{F}_p$$

Generation of a decomposition

Construction of a pool of derived random variables:

$$\forall j \in \{n, \lambda - 1\}, x_j = \phi_j(x_0, \dots, x_{j-1}) = \psi_j\left(\sum_{k=0}^{j-1} \alpha_k \cdot x_k\right)$$

Layout of the decomposition:

$$f(x_0, \dots, x_{n-1}) = \sum_{i=0}^{t-1}\left(\sum_{j=0}^{n+\lambda-1} \beta_{i,j} \cdot x_j\right) \cdot \left(\sum_{k=0}^{n+\lambda-1} d_{i,k} \cdot x_j\right)$$

- $d_{i,k}$: generated at random
- $\beta_{i,j}$: determined by the algorithm

Cost of a decomposition

Homomorphic multiplications can be done with **two** calls of the PBS with function $x \mapsto x^2$.
It comes from the identity: $xy = \frac{1}{4}\left((x+y)^2 - (x-y)^2\right)$.

Cost of a decomposition

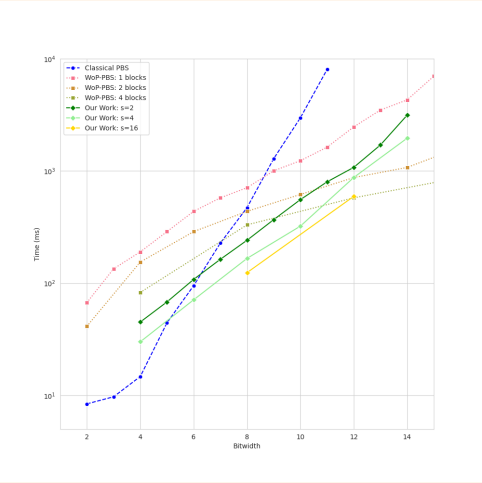
$$\#_{\mathsf{PBS}} = \lambda + 2t$$

Determinations of the β's

$$s^n \left\{ \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ \vdots \\ y_{s^n-1} \end{pmatrix} \right. = \overbrace{\left(\begin{array}{c|c|c|c} \mathcal{A}_0 & \mathcal{A}_1 & \dots & \mathcal{A}_{t-1} \end{array} \right)}^{t\cdot(n+\lambda)} \cdot \left(\begin{array}{c} \beta_{0,0} \\ \vdots \\ \beta_{0,n+\lambda-1} \\ \hline \vdots \\ \hline \beta_{t-1,0} \\ \vdots \\ \beta_{t-1,n+\lambda-1} \end{array} \right)$$

$$\mathcal{A}_i = \begin{pmatrix} x_0^{(0)} \cdot \left\langle \vec{d}_i, \vec{x}^{(0)} \right\rangle & \dots & x_{n+\lambda-1}^{(0)} \cdot \left\langle \vec{d}_i, \vec{x}^{(0)} \right\rangle \\ x_0^{(1)} \cdot \left\langle \vec{d}_i, \vec{x}^{(1)} \right\rangle & \dots & x_{n+\lambda-1}^{(1)} \cdot \left\langle \vec{d}_i, \vec{x}^{(1)} \right\rangle \\ \vdots & \vdots & \vdots \\ x_0^{(s^n-1)} \cdot \left\langle \vec{d}_i, \vec{x}^{(s^n-1)} \right\rangle & \dots & x_{n+\lambda-1}^{(s^n-1)} \cdot \left\langle \vec{d}_i, \vec{x}^{(s^n-1)} \right\rangle \end{pmatrix}$$

Performances



Contributions

Transciphering with Transistor

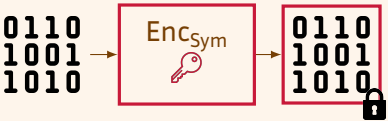
A solution to ciphertext expansion: Transciphering

Client

Server

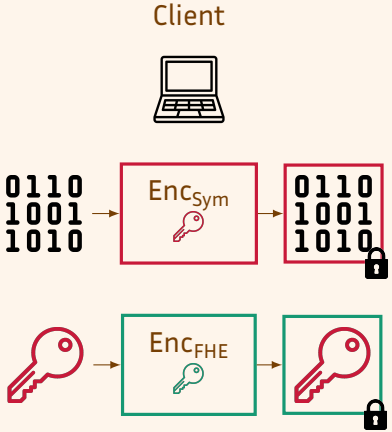
A solution to ciphertext expansion: Transciphering

Client

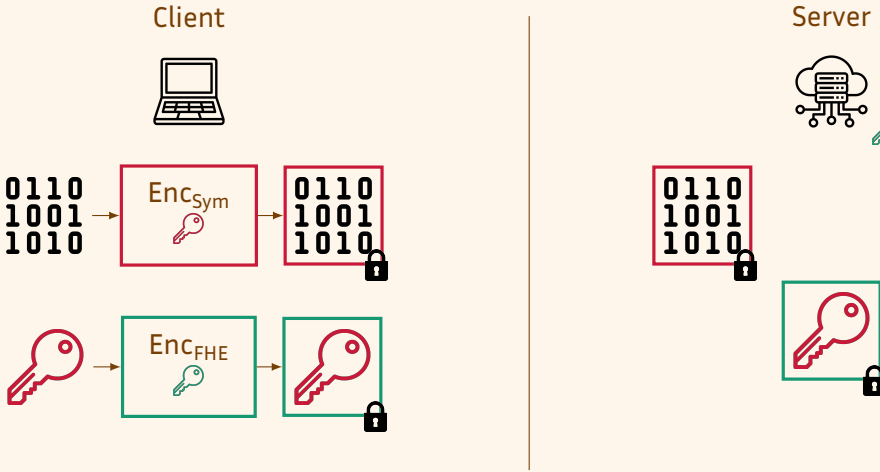


Server

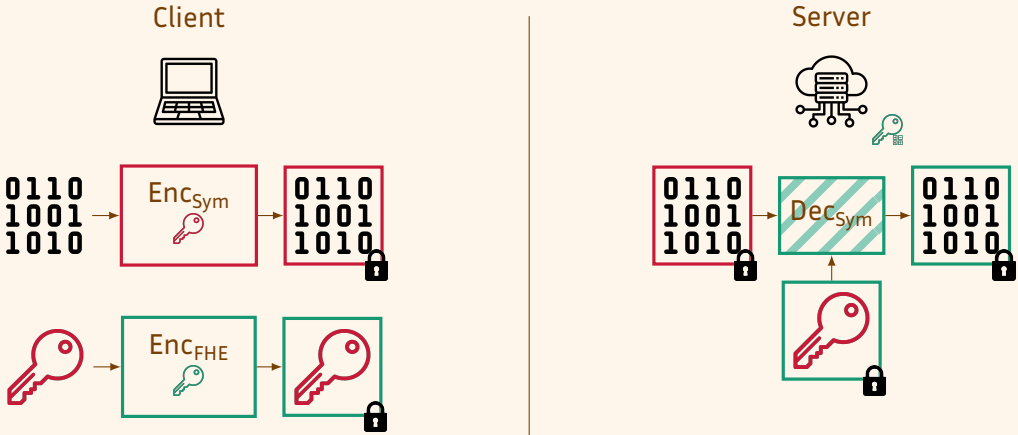
A solution to ciphertext expansion: Transciphering



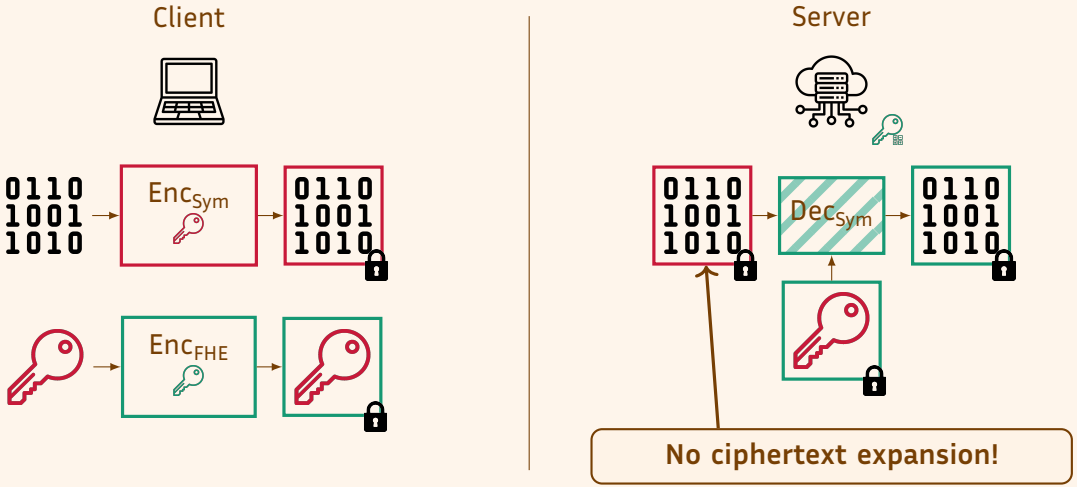
A solution to ciphertext expansion: Transciphering



A solution to ciphertext expansion: Transciphering



A solution to ciphertext expansion: Transciphering

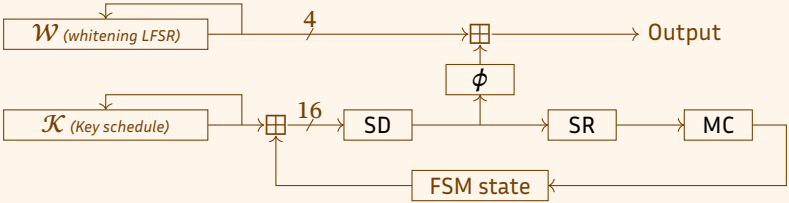


A solution to ciphertext expansion: Transciphering

We propose a new symmetric cipher, that shows good performances in the homomorphic domain.

Design of Transistor

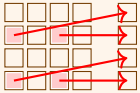
Prime field: $\mathbb{F}_{17}$



(a) SD.

(b) SR.

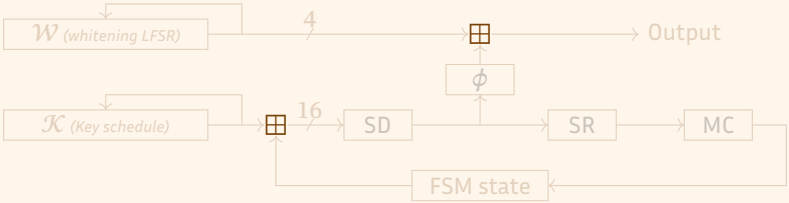
(c) MC.



(d) ϕ.

Design of Transistor

Prime field: $\mathbb{F}_{17}$



(a) SD.

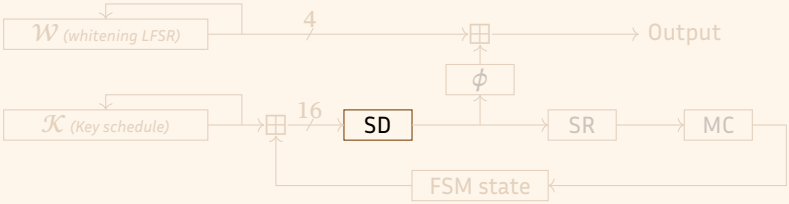
(b) SR.

(c) MC.

(d) ϕ.

Design of Transistor

Prime field: $\mathbb{F}_{17}$



(a) SD.

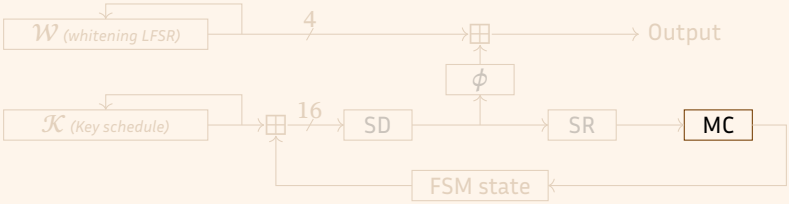
(b) SR.

(c) MC.

(d) ϕ.

Design of Transistor

Prime field: $\mathbb{F}_{17}$



(a) SD.

(b) SR.

(c) MC.

(d) ϕ.

MixColumns

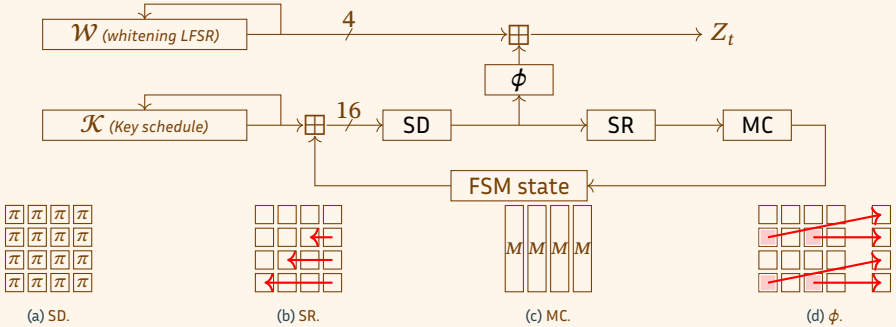
The matrix we chose for MixColumns is:

$$M = \begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & -1 & 1 & -2 \\ 1 & 1 & -2 & -1 \\ 1 & -2 & -1 & 1 \end{bmatrix}.$$

- Matrix MDS to ensure optimal diffusion,
- Symmetric,
- Minimal ℓ_2-norm of 7 $\rightarrow$ important for noise management.

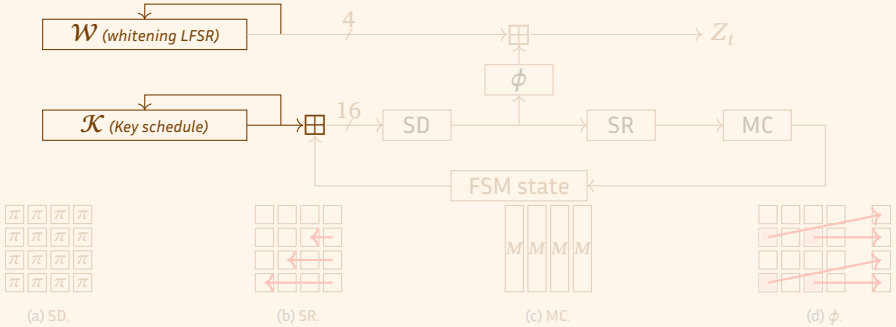
Design of Transistor

Prime field: $\mathbb{F}_{17}$

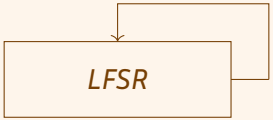


Design of Transistor

Prime field: $\mathbb{F}_{17}$

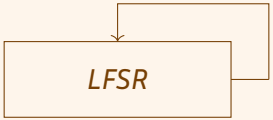


Silent LFSR



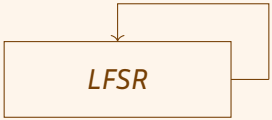
- Naive approach : linearly update the state at each clock.

Silent LFSR



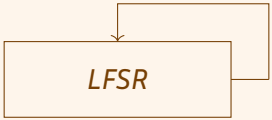
- Naive approach : linearly update the state at each clock.
- Problem: the noise accumulates over time.

Silent LFSR



- Naive approach : linearly update the state at each clock.
- Problem: the noise accumulates over time.
- Solution: Computing on the fly the coefficients of the linear combination in clear

Silent LFSR



- Naive approach : linearly update the state at each clock.
- Problem: the noise accumulates over time.
- Solution: Computing on the fly the coefficients of the linear combination in clear

The noise variance in the output of the silent LFSR remains stable over time, without using any PBS.

Performances

Cipher	Setup	Latency	Throughput	Communication Cost[a]	p_{err}
`Trivium` [BOS23] (128 thr.)	2259 ms	121 ms	529 bits/s	640 B + 35.6 MB †	2^{-40}
`Kreyvium` [BOS23] (128 thr.)	2883 ms	150 ms	427 bits/s	1024 B + 35.6 MB †	2^{-40}
`Margrethe` [HMS23]	No	27.2 ms	147.06 bits/s	64 MB *	$< 2^{-1000}$
	No	54.2 ms	73.8 bits/s	128 MB *	$< 2^{-1000}$
PRF-based construction [DJL+24]	No	5.675 ms	881 bits/s	32.8 MB = 8.9 MB + 23.9 MB	2^{-64}
FRAST [CCH+24]	25 s (8 thr.)	6.2 s	20.66 bits/s	34.05 MB = 148 KB + 33.91 MB	2^{-80}
`Transistor`	No	251 ms	65.10 bits/s	13.54 MB = 780 B + 12.78 MB	2^{-128}

[a] Includes size of encrypted symmetric key + size of evaluation keys. † Values recomputed from the data of the papers. For consistency's sake, we applied the classical technique of ciphertexts compression to estimate the communication cost.

* In `Margrethe`, no keyswitching nor bootstrapping keys are required.

Part 3

Conclusion

Conclusion

Studying plaintext spaces not power of two yielded speed-ups in different use-cases:

- Acceleration of boolean functions
- Acceleration of large LUT
- Improvement of transciphering performances

Perspectives

- More study of those “exotic” plaintext spaces
- Can we do this kind of things with packed schemes such as CKKS ?

Bibliography I


Thibault Balenbois, Jean-Baptiste Orfila, and Nigel P. Smart.
Trivial transciphering with trivium and TFHE.
In Michael Brenner, Anamaria Costache, and Kurt Rohloff, editors, *Proceedings of the 11th Workshop on Encrypted Computing & Applied Homomorphic Cryptography, Copenhagen, Denmark, 26 November 2023*, pages 69–78. ACM, 2023.


Adda-Akram Bendoukha, Oana Stan, Renaud Sirdey, Nicolas Quero, and Luciano Freitas.
Practical homomorphic evaluation of block-cipher-based hash functions with applications.
Cryptology ePrint Archive, Report 2023/480, 2023.

Bibliography II

- Mingyu Cho, Woohyuk Chung, Jincheol Ha, Jooyoung Lee, Eun-Gyeol Oh, and Mincheol Son.
  FRAST: TFHE-friendly cipher based on random S-boxes.
  IACR Trans. Symm. Cryptol., 2024(3):1–43, 2024.
- Jean-Sébastien Coron, Tancrède Lepoint, and Mehdi Tibouchi.
  Scale-invariant fully homomorphic encryption over the integers.
  In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 311–328. Springer, Berlin, Heidelberg, March 2014.
- Amit Deo, Marc Joye, Benoit Libert, Benjamin R. Curtis, and Mayeul de Bellabre.
  Fast homomorphic evaluation of LWR-based PRFs.
  Cryptology ePrint Archive, Paper 2024/665, 2024.

Bibliography III


Craig Gentry.
Fully homomorphic encryption using ideal lattices.
In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.


Craig Gentry, Shai Halevi, and Nigel P. Smart.
Homomorphic evaluation of the AES circuit.
In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 850–867. Springer, Berlin, Heidelberg, August 2012.

Bibliography IV


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. Springer, Cham, December 2023.


Daphné Trama, Pierre-Emmanuel Clet, Aymen Boudguiga, and Renaud Sirdey.
A homomorphic AES evaluation in less than 30 seconds by means of TFHE.
In Michael Brenner, Anamaria Costache, and Kurt Rohloff, editors, *Proceedings of the 11th Workshop on Encrypted Computing & Applied Homomorphic Cryptography, Copenhagen, Denmark, 26 November 2023*, pages 79–90. ACM, 2023.