Cryptanalysis of Elisabeth-4

Henri Gilbert, Rachelle Heim Boissier, Jérémy Jean, Jean-René Reinhard

Asiacrypt 2023

Introduction

About `Elisabeth-4`

- Stream cipher published at Asiacrypt 2022.
- Designed by Cosseron, Hoffman, Méaux, Standaert.
- Tailored for Fully Homomorphic Encryption (FHE) use cases.
- 128-bit security claim.

Our contribution

- Full break of `Elisabeth-4`.
- Linearisation attack that exploits:
  - Sparsity of the linear system;
  - Rank defects;
  - Filtering strategies.

Hybrid Homomorphic Encryption

Symmetric key K
Hom. key (SK, PK)
Data D

User | **Server**

- Encrypt D under K using symmetric enc algo E
- Encrypt K under PK using homomorphic enc algo E^{hom}

$(E^{hom}_{PK}(K), E_K(D)) \longrightarrow$

- *Transciphering*:
  Transform $E_K(D)$ into $E^{hom}_{PK}(D)$ using $E^{hom}_{PK}(K)$
- Perform computations homomorphically.
  Obtain R.

$\longleftarrow R$

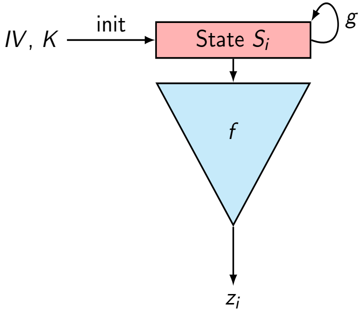
- Decrypt R using SK, and obtains the result of the computation

Symmetric cryptography for FHE

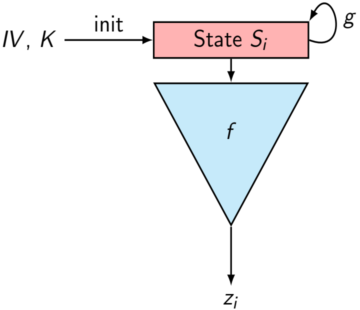
Encryption algorithms for FHE

- Classical symmetric encryption algorithms (e.g. AES): **not efficient** in FHE.
- This led to the design of **new algorithms**:
  Ex: `LowMC` [ARSTZ16], `Kreyvium` [CCFLNPS16], `FLIP` [CMJS16]
- The stream cipher `Elisabeth-4` is a recent example (AC2022).

A classical cryptanalysis technique: Linearisation

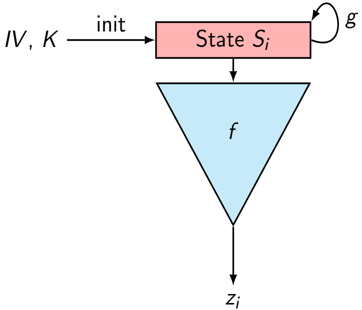


A classical cryptanalysis technique: Linearisation



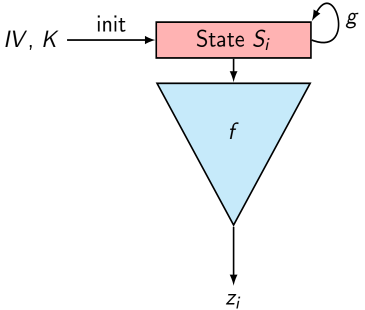
- Consider non-linear equations $z_i = F_i(K_0, \cdots, K_{n-1})$.

A classical cryptanalysis technique: Linearisation



- Consider non-linear equations $z_i = F_i(K_0, \cdots, K_{n-1})$.
- View them as linear equations: view each **monomial** in the key bits as an independant variable.

A classical cryptanalysis technique: Linearisation

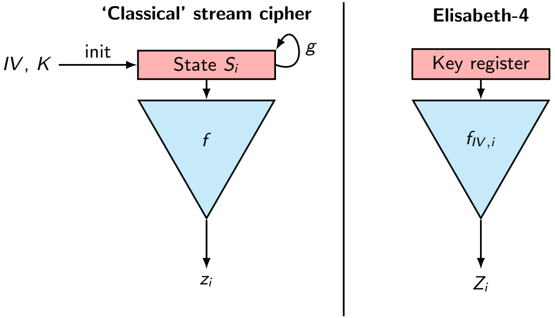


- Consider non-linear equations $z_i = F_i(K_0, \cdots, K_{n-1})$.
- View them as linear equations: view each **monomial** in the key bits as an independant variable.
- Solve the linear system.

Elisabeth-4 FHE-friendly features

Elisabeth-4 has been '*conceived to take advantage of the efficient operations of the FHE scheme* **TFHE**' [CGGI20].

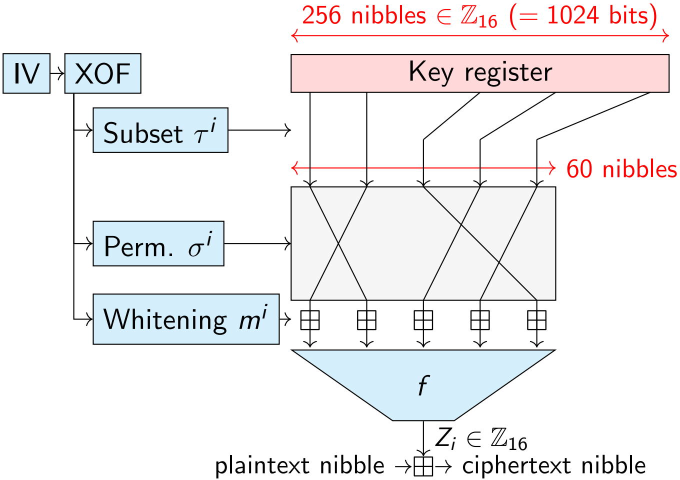
- A slightly different **structure** as compared to other stream ciphers:



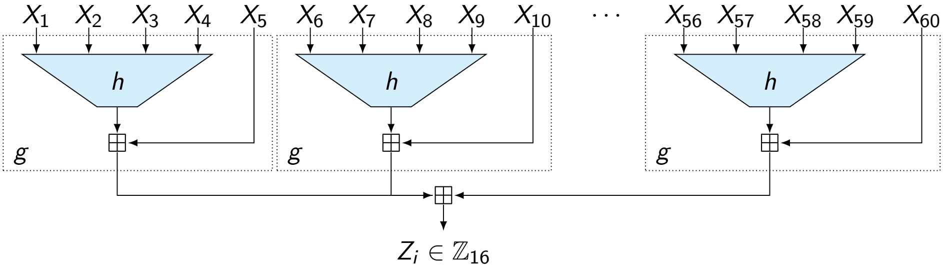
- Specified using operations over $\mathbb{Z}_q$ with $q = 2^4 = 16$.
- Use of *negacyclic look-up tables*: $\forall X \in \mathbb{Z}_{16}$, $S[X + 2^3] = S[-X]$.

Plan

Elisabeth-4: overall structure



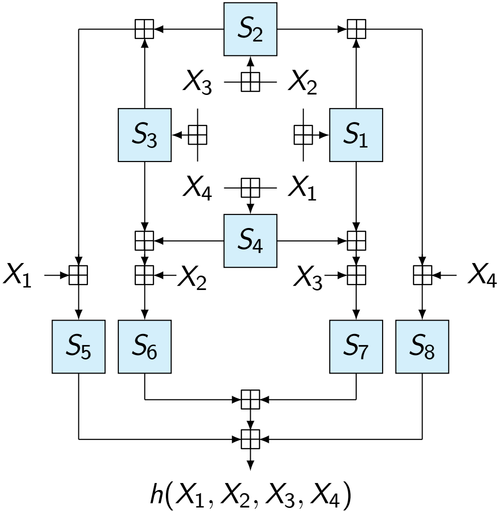
The filtering function f



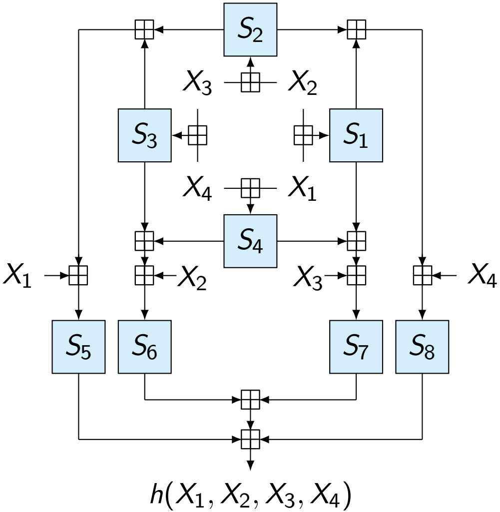
Structure of f

- 12 parallel calls to a 5-to-1 function g.
- $g(X_1, X_2, X_3, X_4, X_5) = h(X_1, X_2, X_3, X_4) + X_5$
- h is non-linear.
  - ingredients: $+$ and negacyclic look-up tables.

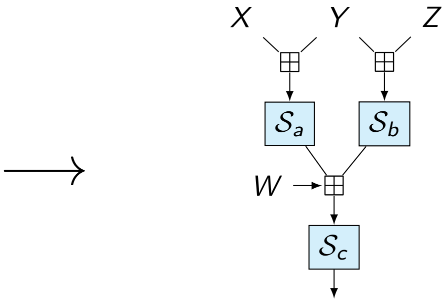
The non-linear function h



The non-linear function h

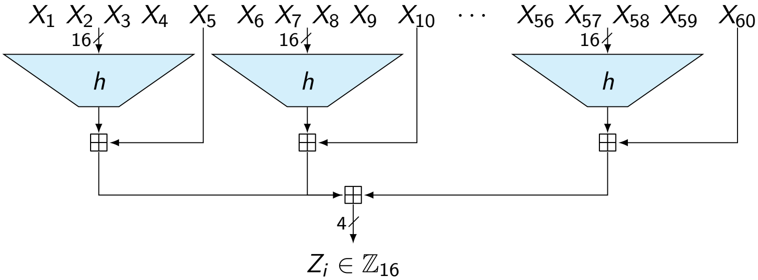


Sum of 4 '*Antler functions*'



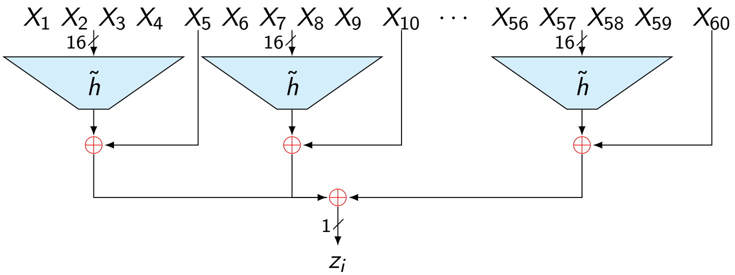
Plan

The filtering function f



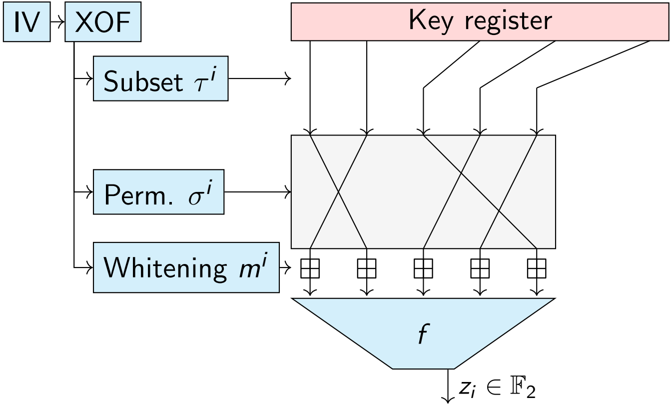
Basic linearisation in $\mathbb{F}_2$

The filtering function f



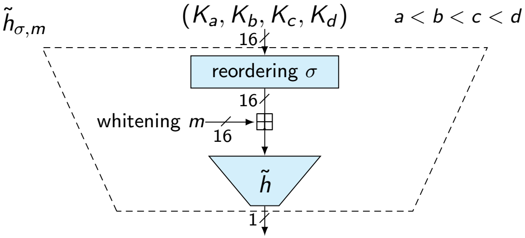
We focus on the **LSB of the output nibble**
$\rightarrow$ On the LSB, the addition in $\mathbb{Z}_{16}$ acts as an XOR.

Basic linearisation in $\mathbb{F}_2$



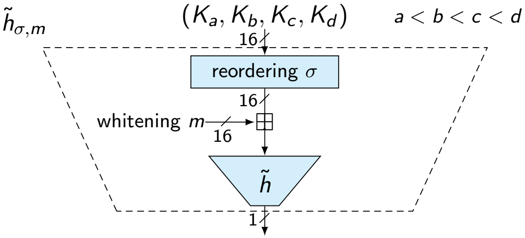
How many monomials can appear in the ANF of the LSB **regardless of the choice of subset/permutation/whitening** ?

Bounding the number of monomials



1. For any 4-tuple $a < b < c < d$ of key register positions, the number of monomials in **all** variations $\tilde{h}_{\sigma,m}$ of h is bounded by 2^{16}.

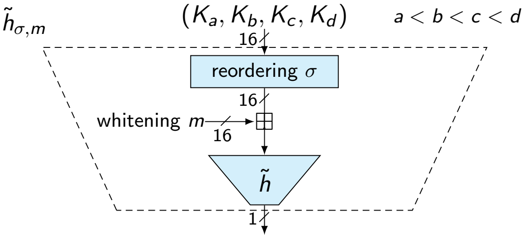
Bounding the number of monomials



1. For any 4-tuple $a < b < c < d$ of key register positions, the number of monomials in **all** variations $\tilde{h}_{\sigma,m}$ of h is bounded by 2^{16}.
2. How many possible choices of (K_a, K_b, K_c, K_d) in the 256-nibble key register?

$$\binom{256}{4}$$

Bounding the number of monomials



1. For any 4-tuple $a < b < c < d$ of key register positions, the number of monomials in **all** variations $\tilde{h}_{\sigma,m}$ of h is bounded by 2^{16}.
2. How many possible choices of (K_a, K_b, K_c, K_d) in the 256-nibble key register?

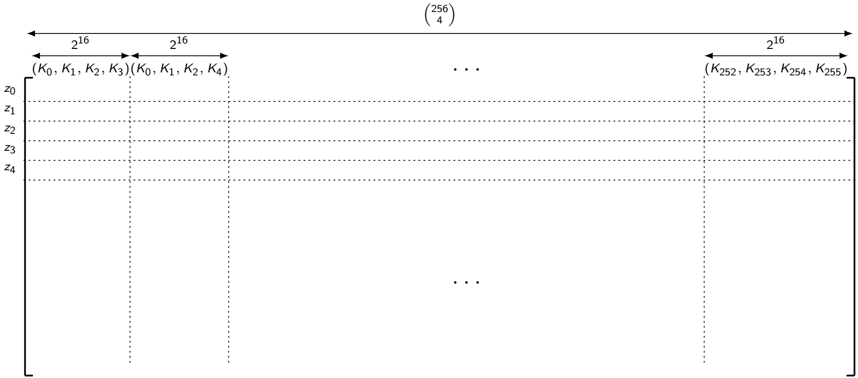
$$\binom{256}{4}$$

Total number of monomials $\leq \mu = \binom{256}{4} 2^{16}$.

Building a linearisation matrix

At each iteration of the stream cipher:

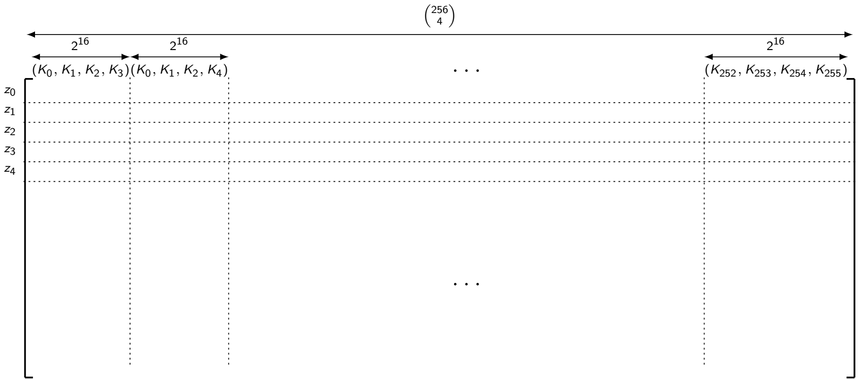
- Build the ANF of the keystream nibble LSB z_i by combining the contribution of every h function.



Building a linearisation matrix

Linearisation matrix A

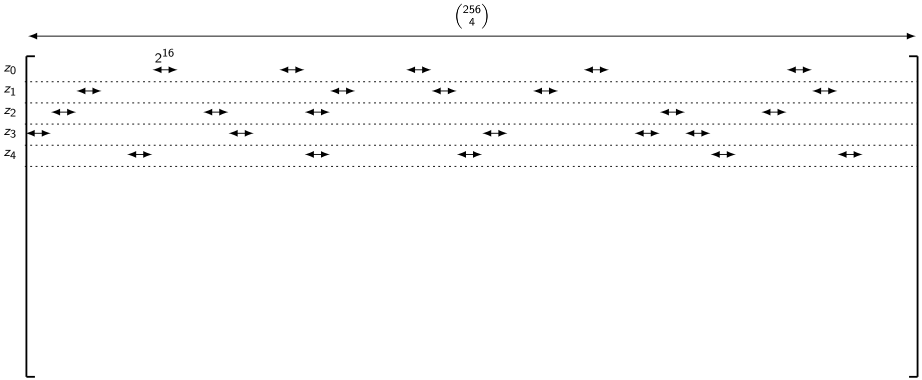
- Each **column** corresponds to a **monomial**: $\binom{256}{4}2^{16} \approx 2^{43.4}$ columns.
- Each set of 2^{16} columns corresponds to the monomials in the bits of (K_a, K_b, K_c, K_d), $a < b < c < d$.



Building a linearisation matrix

At each iteration of the stream cipher, the XOF outputs a subset, a permutation, a whitening vector which determine:

- 12 subsets $\{K_a, K_b, K_c, K_d\}$ associated with a block of 2^{16} columns;
- the ANF for each of these 12 blocks.



Resulting linearisation attack

Basic linearisation attack

- Using $\mu = \binom{256}{4} \cdot 2^{16} \approx 2^{43.4}$ keystream elements' LSB, a solvable linear system is built.
- This linear system is solved in μ^{ω} operations.
  - Straightforward Gaussian elimination, $\omega = 3$, $T \approx 2^{131}$ operations.
- **Data complexity** is μ nibbles.

Resulting linearisation attack

Basic linearisation attack

- Using $\mu = \binom{256}{4} \cdot 2^{16} \approx 2^{43.4}$ keystream elements' LSB, a solvable linear system is built.
- This linear system is solved in μ^{ω} operations.
  - Straightforward Gaussian elimination, $\omega = 3$, $T \approx 2^{131}$ operations.
- **Data complexity** is μ nibbles.

First observation: A is sparse.

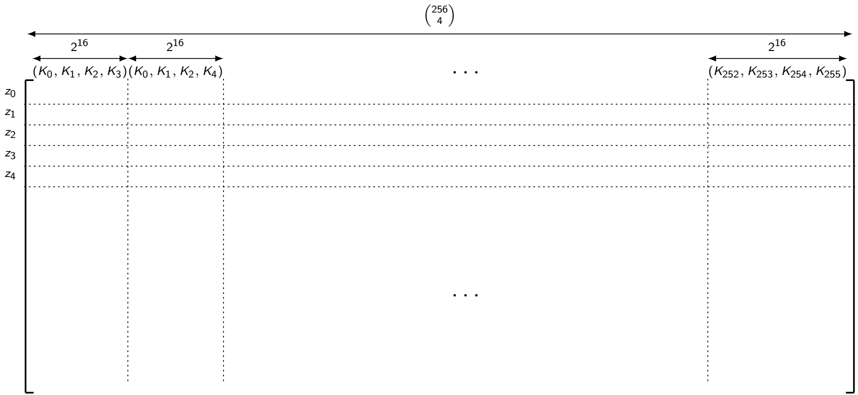
- At most $s = 12 \cdot 2^{16} \ll \mu$ non-zero bits on each row.
- **Memory complexity**: $s \cdot \mu \approx 2^{63}$ bits.
- Sparse linear algebra: **Coppersmith's Block-Wiedemann algorithm**.
  - **Main idea**: only use matrix-vector multiplication, which costs $\mathcal{O}(s \cdot n)$ operations.
- **Improved time complexity:** $\mu^3 \rightarrow \frac{6}{64} \cdot s \cdot \mu^2$.
- $T \approx 2^{103}$ operations.

Plan

Identification of a rank defect

Linearization matrix

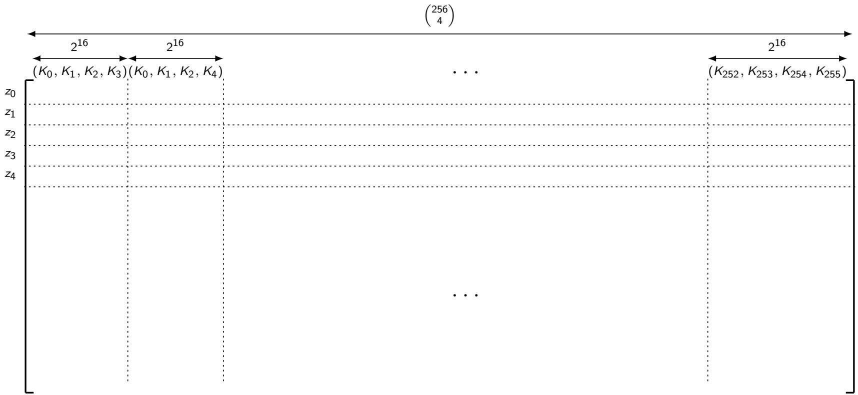
- We show that the linearization matrix has a **rank defect**.



Identification of a rank defect

Linearization matrix

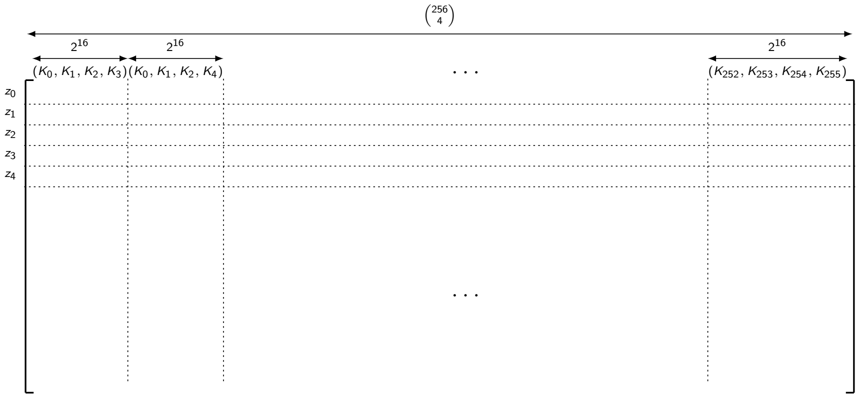
- We show that the linearization matrix has a **rank defect**. How?



Identification of a rank defect

Linearization matrix

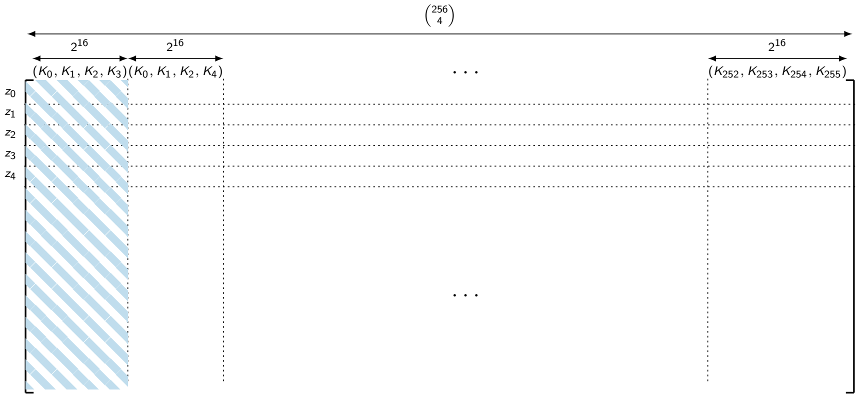
- We show that the linearization matrix has a **rank defect**. How?
- We computed the maximum possible rank of any of the $\binom{256}{4}$ **submatrixes** corresponding to a choice of (K_a, K_b, K_c, K_d), $a < b < c < d$.



Identification of a rank defect

Linearization matrix

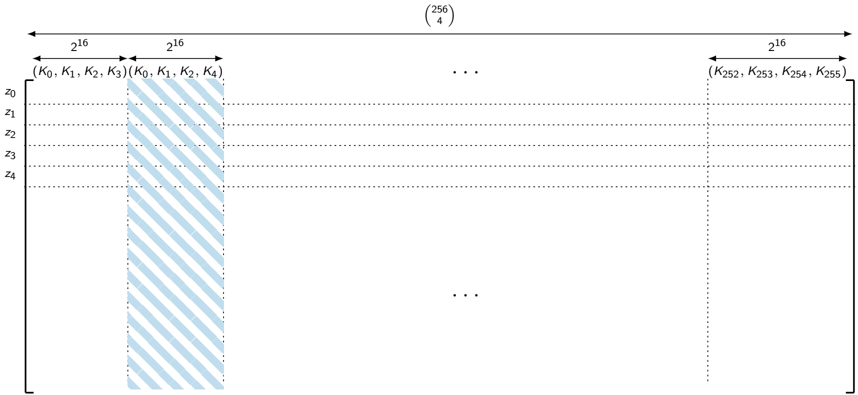
- We show that the linearization matrix has a **rank defect**. How?
- We computed the maximum possible rank of any of the $\binom{256}{4}$ **submatrixes** corresponding to a choice of (K_a, K_b, K_c, K_d), $a < b < c < d$.



Identification of a rank defect

Linearization matrix

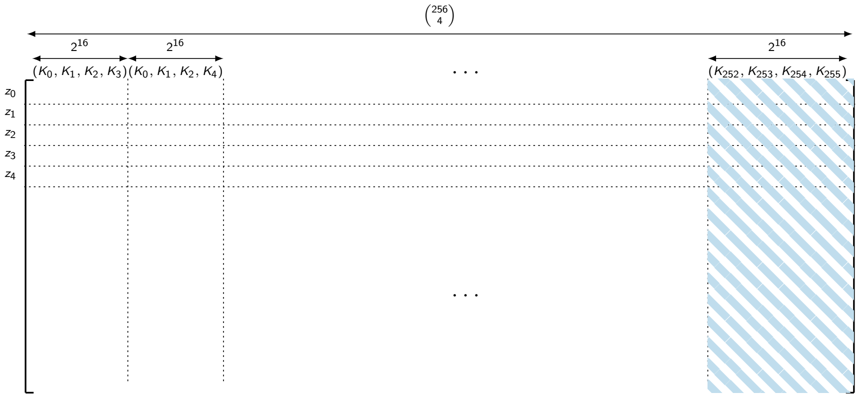
- We show that the linearization matrix has a **rank defect**. How?
- We computed the maximum possible rank of any of the $\binom{256}{4}$ **submatrixes** corresponding to a choice of (K_a, K_b, K_c, K_d), $a < b < c < d$.



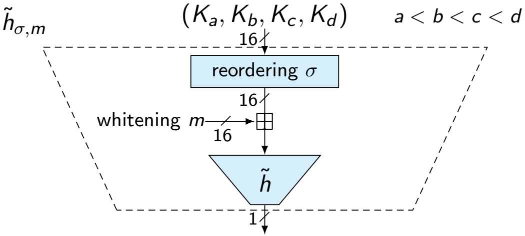
Identification of a rank defect

Linearization matrix

- We show that the linearization matrix has a **rank defect**. How?
- We computed the maximum possible rank of any of the $\binom{256}{4}$ **submatrixes** corresponding to a choice of (K_a, K_b, K_c, K_d), $a < b < c < d$.



Identification of a rank defect



We pre-computed and stored the ANF of $2^{16} \cdot 4!$ *variations* $\tilde{h}_{\sigma,m}$ of h constructed by

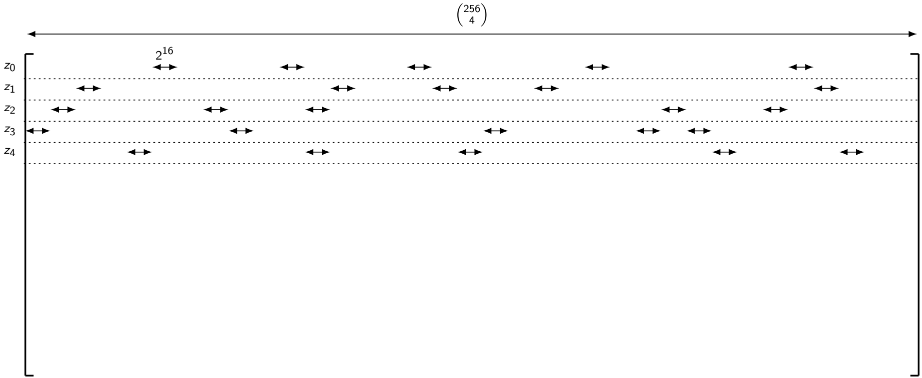
- restricting the output to the LSB;
- considering the 4! **possible orderings of the variables**;
- and the 2^{16} **possible masks**.

We computed the rank and obtained

$$\dim\left(< \tilde{h}_{IV,i} >\right) \leq \dim\left(< \tilde{h}_{M,\sigma} >\right) = \rho = 2^{13.08} \ll 2^{16} .$$

Linearisation matrix

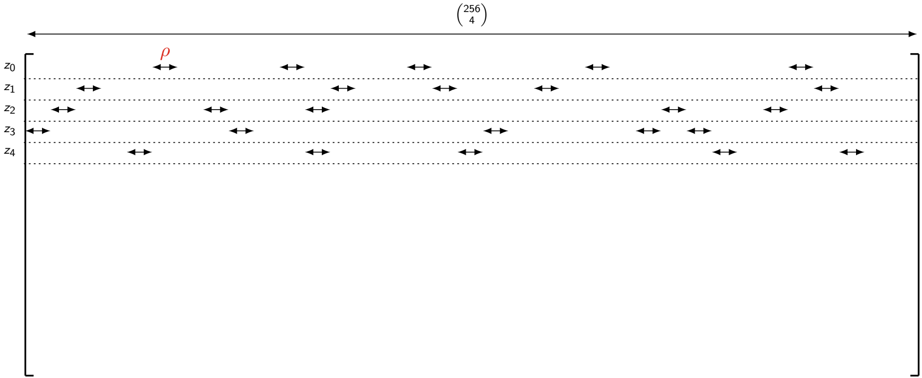
- Basic attack: Each column corresponds to a monomial.
- But, each vector in a block of size 2^{16} can be written in a **basis of size ρ**.



Exploiting the rank defect

Linearisation matrix

- $\mathbf{A}$ has now only $\mu' = \binom{256}{4}\rho$ columns
- Each row has at most $s' = 12 \cdot \rho$ active bits.



Improved attack

- **Time complexity:** $\frac{6}{64} \cdot s \cdot \mu^2$
- **Data complexity:** μ
- **Memory complexity:** $s \cdot \mu$

Improved attack

- **Time complexity:** $\frac{6}{64} \cdot s \cdot \mu^2 \rightarrow \frac{6}{64} \cdot s' \cdot (\mu')^2 \approx 2^{94}$ operations.
- **Data complexity:** μ
- **Memory complexity:** $s \cdot \mu$

Improved attack

- **Time complexity:** $\frac{6}{64} \cdot s \cdot \mu^2 \rightarrow \frac{6}{64} \cdot s' \cdot (\mu')^2 \approx 2^{94}$ operations.
- **Data complexity:** $\mu \rightarrow \mu' = 2^{41}$ nibbles.
- **Memory complexity:** $s \cdot \mu$

Improved attack

- **Time complexity:** $\frac{6}{64} \cdot s \cdot \mu^2 \rightarrow \frac{6}{64} \cdot s' \cdot (\mu')^2 \approx 2^{94}$ operations.
- **Data complexity:** $\mu \rightarrow \mu' = 2^{41}$ nibbles.
- **Memory complexity:** $s \cdot \mu \rightarrow s' \cdot \mu' = 2^{57}$ bits.

Explaining the defect (theoretically)

Our results

- We prove a theoretical bound $2^{14.01}$, with $\rho = 2^{13.08} < 2^{14.01} \ll 2^{16}$.
- We also identify and *fully prove* a **degree** defect:

$$\text{For any } IV, i, \ \deg\left(\tilde{h}_{IV,i}\right) \leq 12 < 16\,.$$

.

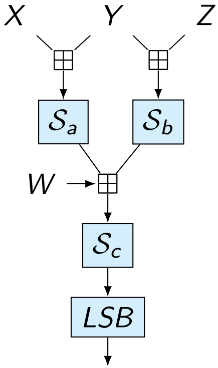
Explaining the defect (theoretically)

Our results

- We prove a theoretical bound $2^{14.01}$, with $\rho = 2^{13.08} < 2^{14.01} \ll 2^{16}$.
- We also identify and *fully prove* a **degree** defect:

$$\text{For any } IV, i, \;\; \deg\left(\tilde{h}_{IV,i}\right) \leq 12 < 16\,.$$

.



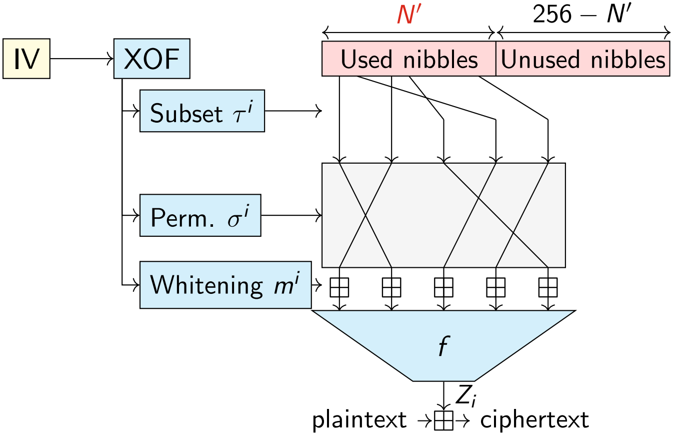
Our analysis (about 1/3 of the article...)

- The rank and degree defects are caused by **HHE-friendly features**.
- Interaction between
  - Negacyclic look-up tables;
  - Addition in $\mathbb{Z}_{16}$.

  within **Antler functions**.

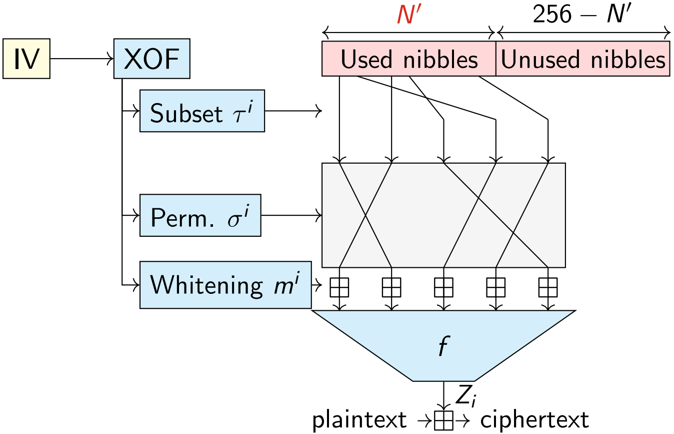
Plan

Considering only convenient XOF outputs



Total number of monomials: $\binom{\cancel{256}}{4}\rho$. .

Considering only convenient XOF outputs



- **Total number of monomials:** $\mu_{N'} = \binom{N'}{4}\rho$.

Chosen-IV attack

- Pre-compute convenient IVs, then query these IVs only.
- The nibbles are all selected in a subset of size N' with probability $p_{N'} \approx \binom{N'}{48} / \binom{256}{48} \rightarrow$ **precomputation cost**: $\mu_{N'} / p_{N'}$ nibbles.
- **Time complexity**: $\frac{6}{64} \cdot s' \cdot (\mu')^2 \rightarrow \lceil \frac{256}{N'} \rceil \cdot \frac{6}{64} \cdot s' \cdot (\mu_{N'})^2 + \mu_{N'} / p_{N'}$ operations.

Chosen-IV attack

- Pre-compute convenient IVs, then query these IVs only.
- The nibbles are all selected in a subset of size N' with probability $p_{N'} \approx \binom{N'}{48}/\binom{256}{48} \rightarrow$ **precomputation cost**: $\mu_{N'}/p_{N'}$ nibbles.
- **Time complexity**: $\frac{6}{64} \cdot s' \cdot (\mu')^2 \rightarrow \lceil \frac{256}{N'} \rceil \cdot \frac{6}{64} \cdot s' \cdot (\mu_{N'})^2 + \mu_{N'}/p_{N'}$ operations.

Trade-off: $N' = 137$.

Chosen-IV attack

- Pre-compute convenient IVs, then query these IVs only.
- The nibbles are all selected in a subset of size N' with probability $p_{N'} \approx \binom{N'}{48} / \binom{256}{48} \rightarrow$ **precomputation cost**: $\mu_{N'}/p_{N'}$ nibbles.
- **Time complexity**: $\frac{6}{64} \cdot s' \cdot (\mu')^2 \rightarrow \lceil \frac{256}{N'} \rceil \cdot \frac{6}{64} \cdot s' \cdot (\mu_{N'})^2 + \mu_{N'}/p_{N'}$ operations.

Trade-off: $N' = 137$. Thus:

- **Time complexity**: $2^{94} \rightarrow 2^{88}$ operations.
- **Data complexity**: $\mu' = 2^{41} \rightarrow \mu_{N'} = 2^{37}$ nibbles.
- **Memory complexity**: $s' \cdot \mu' = 2^{57} \rightarrow s' \cdot \mu_{N'} = 2^{54}$ bits.

Chosen-IV attack

- Pre-compute convenient IVs, then query these IVs only.
- The nibbles are all selected in a subset of size N' with probability $p_{N'} \approx \binom{N'}{48}/\binom{256}{48} \rightarrow$ **precomputation cost**: $\mu_{N'}/p_{N'}$ nibbles.
- **Time complexity**: $\frac{6}{64} \cdot s' \cdot (\mu')^2 \rightarrow \lceil \frac{256}{N'} \rceil \cdot \frac{6}{64} \cdot s' \cdot (\mu_{N'})^2 + \mu_{N'}/p_{N'}$ operations.

Trade-off: $N' = 137$. Thus:

- **Time complexity**: $2^{94} \rightarrow 2^{88}$ operations.
- **Data complexity**: $\mu' = 2^{41} \rightarrow \mu_{N'} = 2^{37}$ nibbles.
- **Memory complexity**: $s' \cdot \mu' = 2^{57} \rightarrow s' \cdot \mu_{N'} = 2^{54}$ bits.

Known-IV attack: Get keystream nibbles until you find enough convenient XOF outputs.

- **Data complexity**: $\mu' = 2^{41} \rightarrow \mu_{N'}/p_{N'} = 2^{87}$ nibbles.

Plan

1 Description of `Elisabeth-4`

2 Basic linearisation

3 Exploiting a rank defect phenomenon

4 Filtering collected equations

5 Small-scale experiments

Small-scale experiments

`https://github.com/jj-anssi/asiacrypt2023-cryptanalysis-elisabeth4`

Toy `Elisabeth-4`

- Operates on $\mathbb{Z}_8$ rather than $\mathbb{Z}_{16}$.
- **Subset** selects 2 sets of key elements among 32 rather than 12 among 256.
- Still has a rank defect, with $\rho = 254 \ll 2^{12}$.

Small-scale experiments

`https://github.com/jj-anssi/asiacrypt2023-cryptanalysis-elisabeth4`

Toy `Elisabeth-4`

- Operates on $\mathbb{Z}_8$ rather than $\mathbb{Z}_{16}$.
- **Subset** selects 2 sets of key elements among 32 rather than 12 among 256.
- Still has a rank defect, with $\rho = 254 \ll 2^{12}$.

Implemented attack

- Two main things we checked:
  - Block-Wiedemann allows to solve an `Elisabeth-4` type linear system.
  - Solving the system allows to recover the key.
- BW implem. from CADO-NFS project for integer factorization.
- Our chosen IV attack using $N' = 12$ required about 35 minutes.

Conclusion

While we did not attempt to patch Elisabeth-4, we believe some tweaks would suffice to prevent our attacks, e.g.:

- larger $r-1$ (larger number of inputs to the h function);
- and/or larger S-box size;
- and/or larger key size.

Conclusion

While we did not attempt to patch Elisabeth-4, we believe some tweaks would suffice to prevent our attacks, e.g.:

- larger $r - 1$ (larger number of inputs to the h function);
- and/or larger S-box size;
- and/or larger key size.

The authors proposed a patch, **check out their paper!** (to appear, Indocrypt 2023)

Conclusion

While we did not attempt to patch Elisabeth-4, we believe some tweaks would suffice to prevent our attacks, e.g.:

- larger $r - 1$ (larger number of inputs to the h function);
- and/or larger S-box size;
- and/or larger key size.

The authors proposed a patch, **check out their paper!** (to appear, Indocrypt 2023)

Thank you for your attention!