A Side-Channel Assisted Cryptanalytic Attack Against **QcBits**

Mélissa Rossi · Mike Hamburg · Michael Hutter · Mark E. Marson

Post-Quantum Cryptography

Quantum computers may threaten the mathematical problems on which public key algorithms are currently based.

$\Rightarrow$ Call for the standardization and transition to post-quantum public key algorithms in the near future

- National Institute of Standards and Technology (NIST)
- European Initiative PQCRYPTO and SAFECRYPTO

Post-Quantum Cryptography

Quantum computers may threaten the mathematical problems on which public key algorithms are currently based.

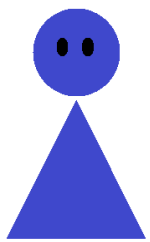
$\Rightarrow$ Call for the standardization and transition to post-quantum public key algorithms in the near future

- National Institute of Standards and Technology (NIST)
- European Initiative PQCRYPTO and SAFECRYPTO

Possible path for post-quantum security

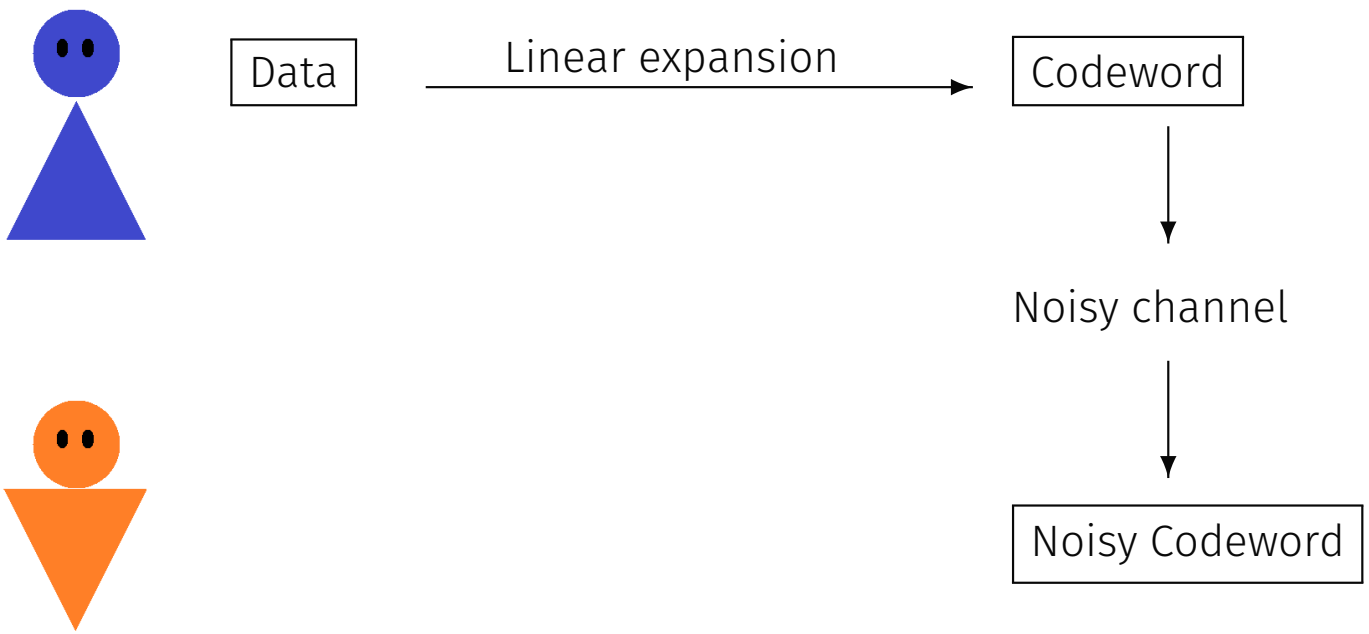
- Error-correcting codes

A binary linear code is a **linear subspace of** $\mathbb{F}_2^n$

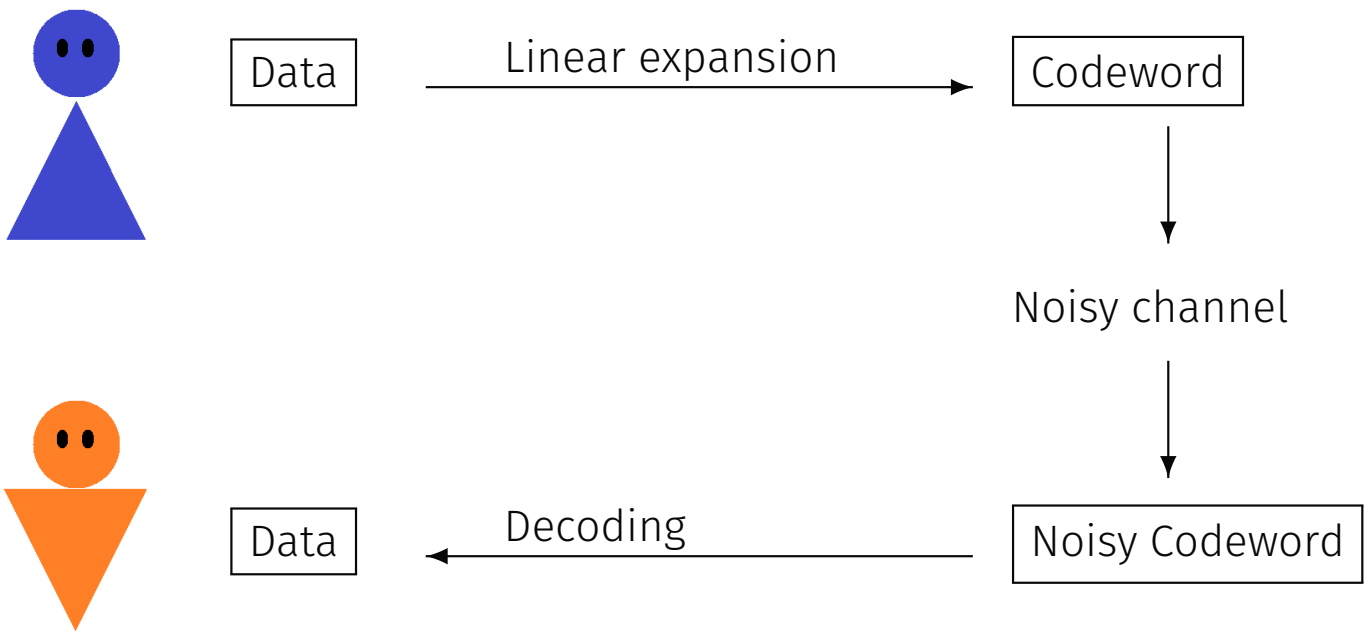


Data → Linear expansion → Codeword

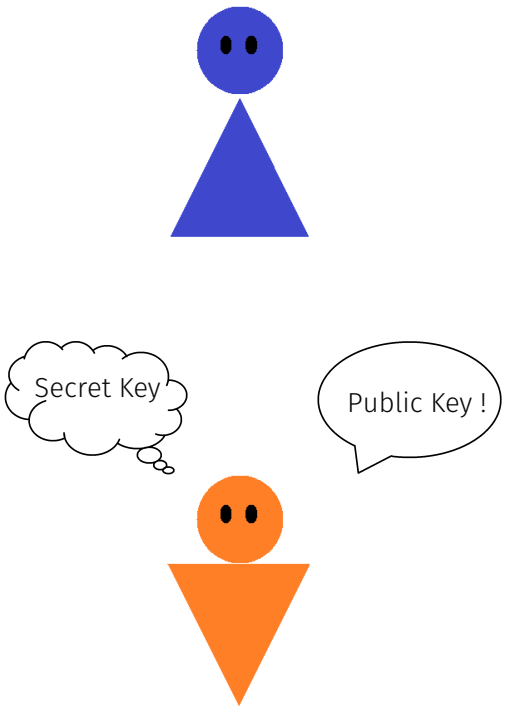
A binary linear code is a **linear subspace of** $\mathbb{F}_2^n$



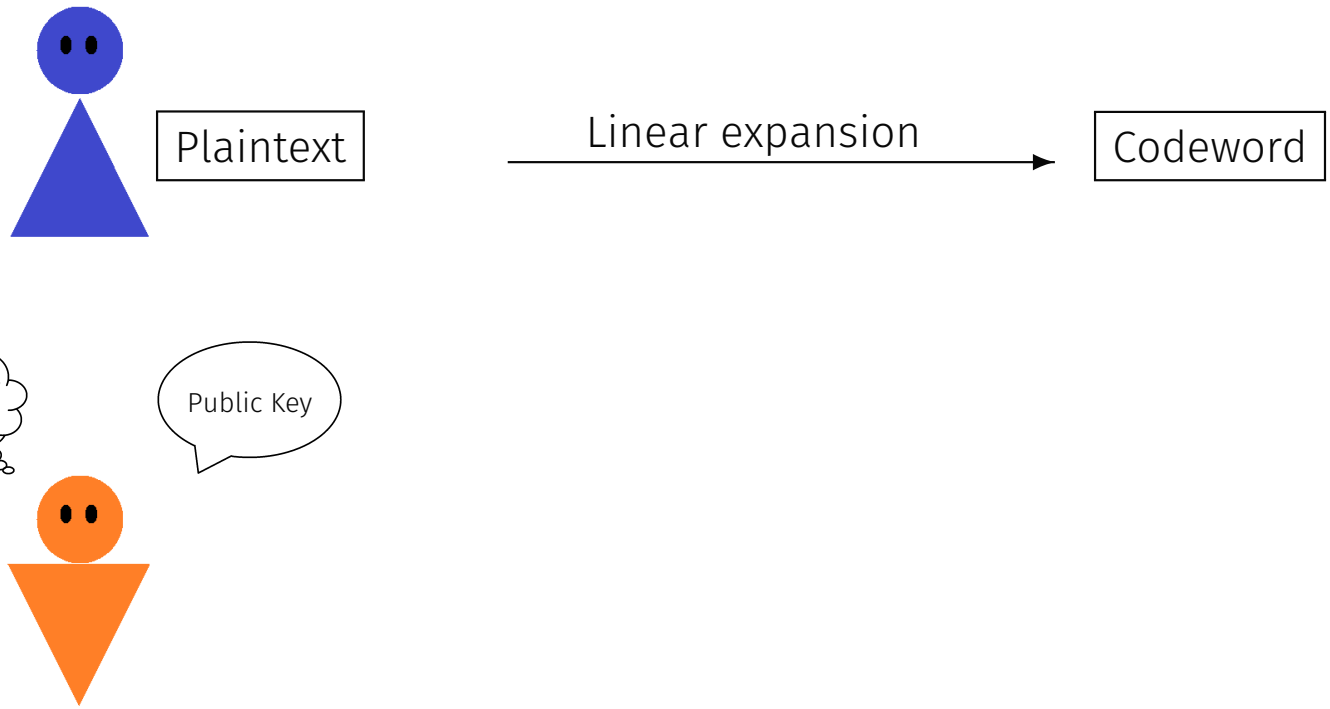
A binary linear code is a **linear subspace of** $\mathbb{F}_2^n$



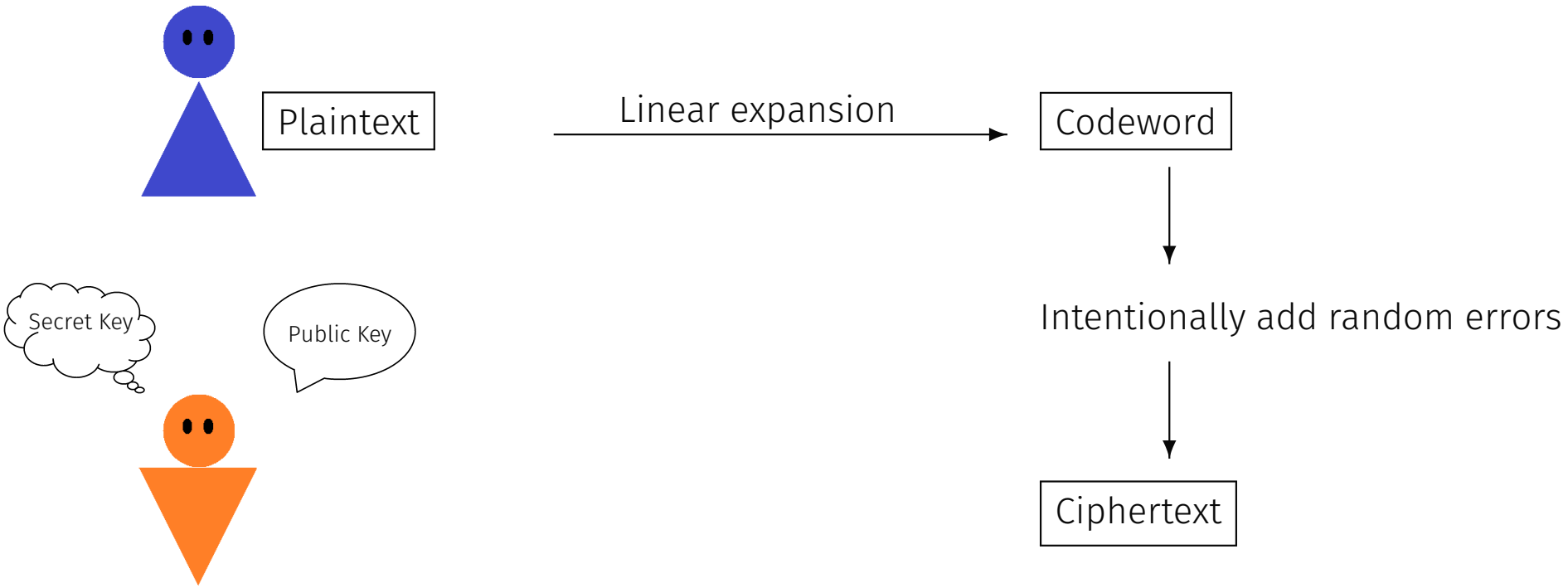
Public key : a way to create a codeword
Secret key : a way to remove the errors



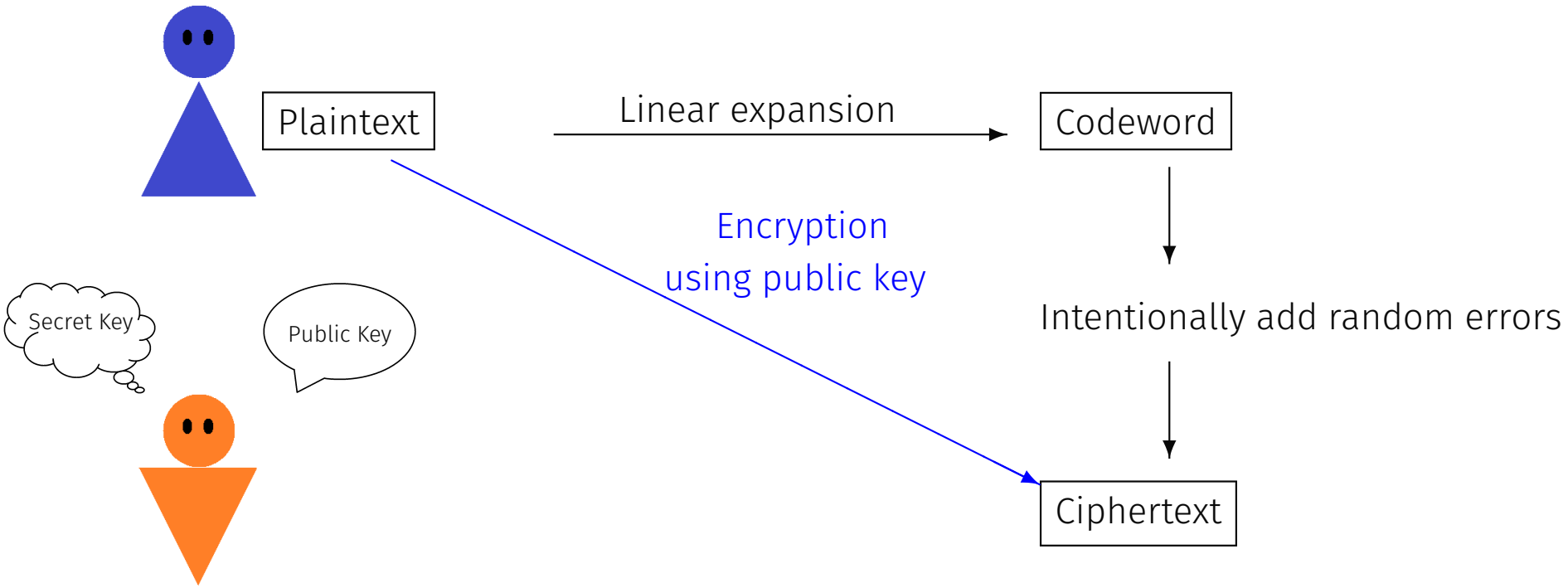
Public key : a way to create a codeword
Secret key : a way to remove the errors



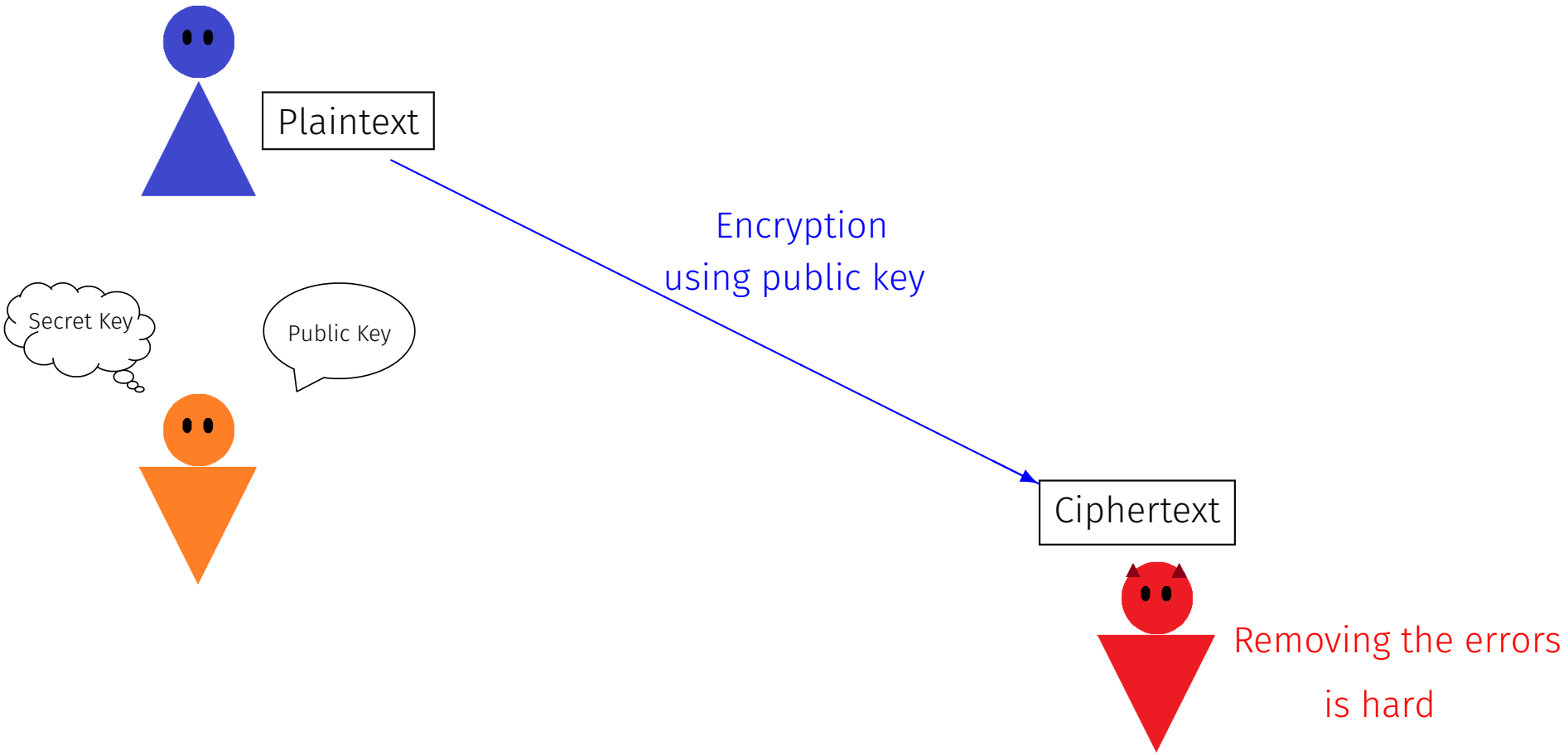
Public key : a way to create a codeword
Secret key : a way to remove the errors



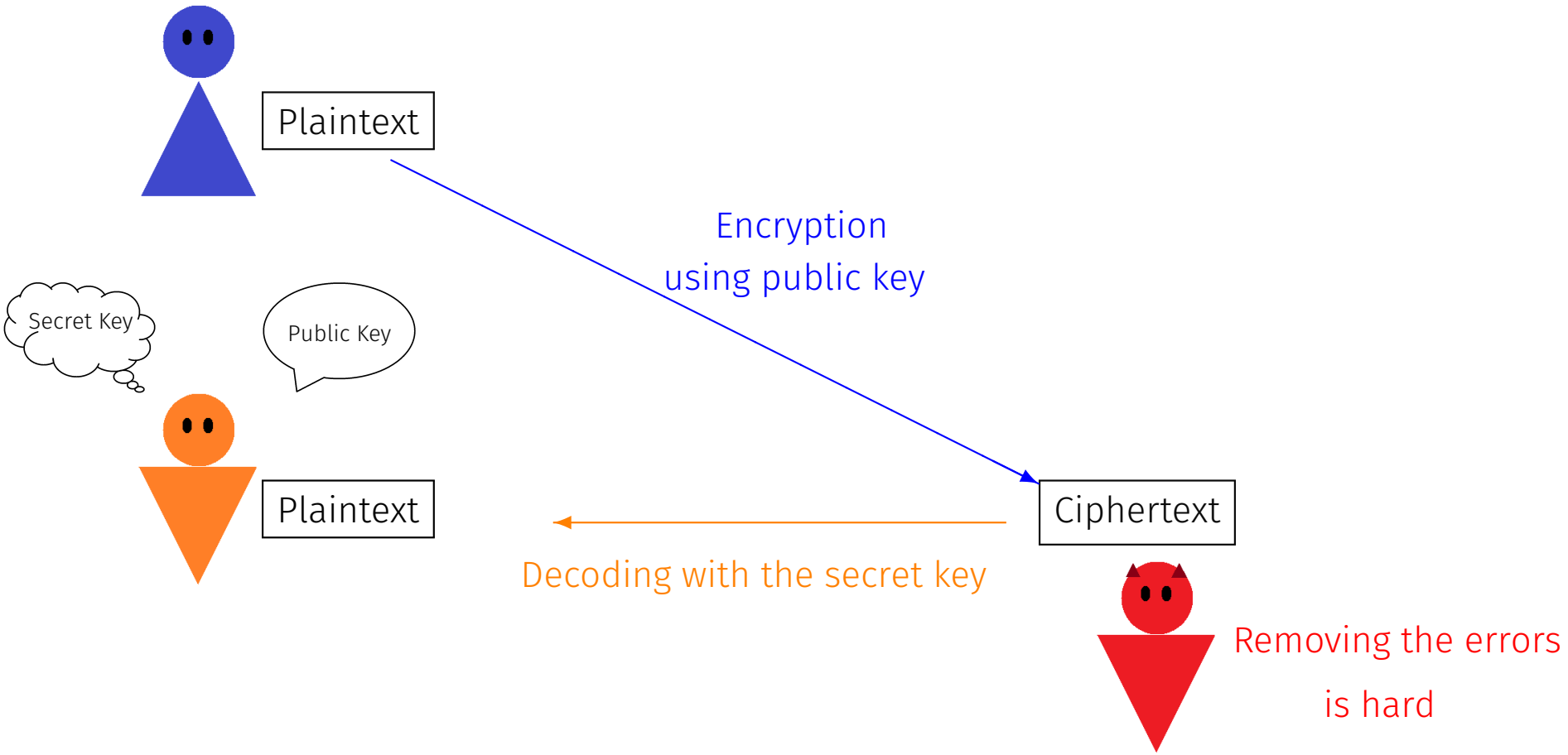
Public key : a way to create a codeword
Secret key : a way to remove the errors



Public key : a way to create a codeword
Secret key : a way to remove the errors



Public key : a way to create a codeword
Secret key : a way to remove the errors



Several possibilities for choosing an appropriate code structure

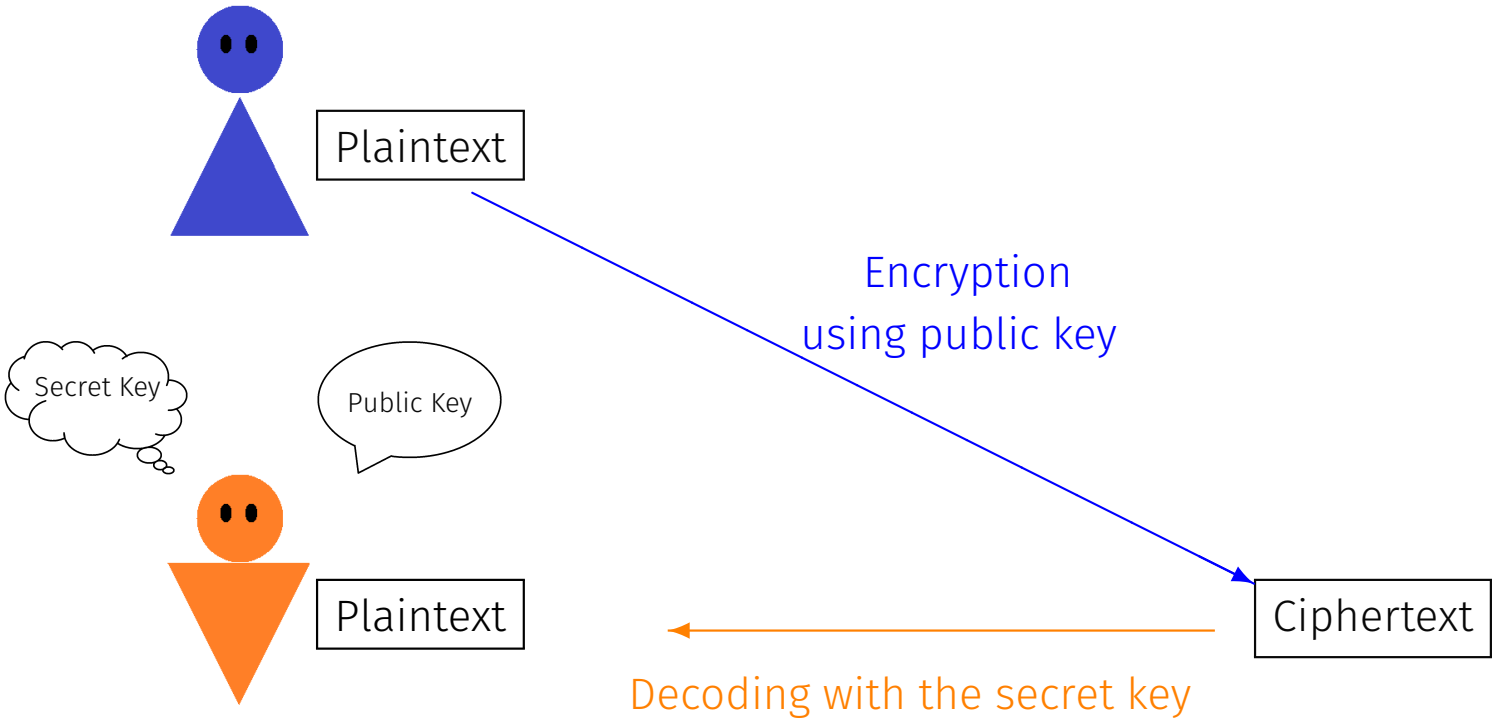
Family of codes	Proposed by	Attacked by
Binary Goppa	original proposition (78)	
Reed Solomon	Niederreiter (86)	Sidelnikov et al (92)
Concatenated	Niederreiter (86)	Sendrier (98)
Reed Muller	Sidelnikov (94)	Minder et al (07)
Algebraic Geometric	Janwa et al(96)	Faure et al (08) Couvreur et al (14)
LDPC	Monico et al (00)	Monico et al (00)
Convolutional	Londahl et al (12)	Landais et al (13)
Wild Goppa	Bernstein et al (10)	Couvreur et al (14) Faugère et al (14)

Several possibilities for choosing an appropriate code structure

Family of codes	Proposed by	Attacked by
Binary Goppa	original proposition (78)	
Reed Solomon	Niederreiter (86)	Sidelnikov et al (92)
Concatenated	Niederreiter (86)	Sendrier (98)
Reed Muller	Sidelnikov (94)	Minder et al (07)
Algebraic Geometric	Janwa et al(96)	Faure et al (08) Couvreur et al (14)
LDPC	Monico et al (00)	Monico et al (00)
Convolutional	Londahl et al (12)	Landais et al (13)
Wild Goppa	Bernstein et al (10)	Couvreur et al (14) Faugère et al (14)
QC MDPC	Misoczki et al (13)	

DESCRIPTION OF QCBITS ALGORITHM

QcBits : A QC MDPC McEliece implementation



Tung Chou, *QcBits: Constant-Time Small-Key Code-Based Cryptography* CHES 2016

- Very fast
- Small key sizes
- Protected against one type of side channel attacks : timing attacks
- 2 sets of parameters : 80 bits and 128 bits security

Size (r)	Hamming weight(w)	Bits of Security
4801	90	80
9857	142	128

Secret key : a QC MDPC matrix H
Public key : a matrix P

$$H = (H_0, H_1)$$

Size (r)	Hamming weight(w)	Bits of Security
4801	90	80
9857	142	128

Secret key : a QC MDPC matrix H
Public key : a matrix P

$$H = (H_0, H_1)$$

$$H = \left(\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \right)$$

Size (r)	Hamming weight(w)	Bits of Security
4801	90	80
9857	142	128

Secret key : a QC MDPC matrix H
Public key : a matrix P

$$H = (H_0, H_1)$$

$$H = \left(\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \right)$$

Quasi Cyclic Moderate Density Parity Check means :

- H_0 and $H_1 \in \mathbb{F}_2^{r \cdot r}$ are circulant
- H_0 and H_1 have sparse rows : only $\frac{w}{2}$ ones
- The codewords x are all the vectors in the right nullspace of H ie $H \cdot x^T = 0$

QcBits : A QC MDPC McEliece implementation

Size (r)	Hamming weight(w)	Bits of Security
4801	90	80
9857	142	128

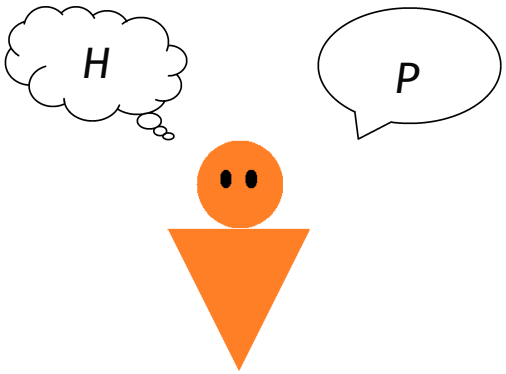
Secret key : a QC MDPC matrix H
Public key : a matrix P

$$H = (H_0, H_1)$$

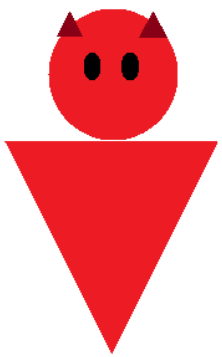
$$P = H_1^{-1} H_0$$

P is circulant too
P is dense

QcBits : Our attacker model

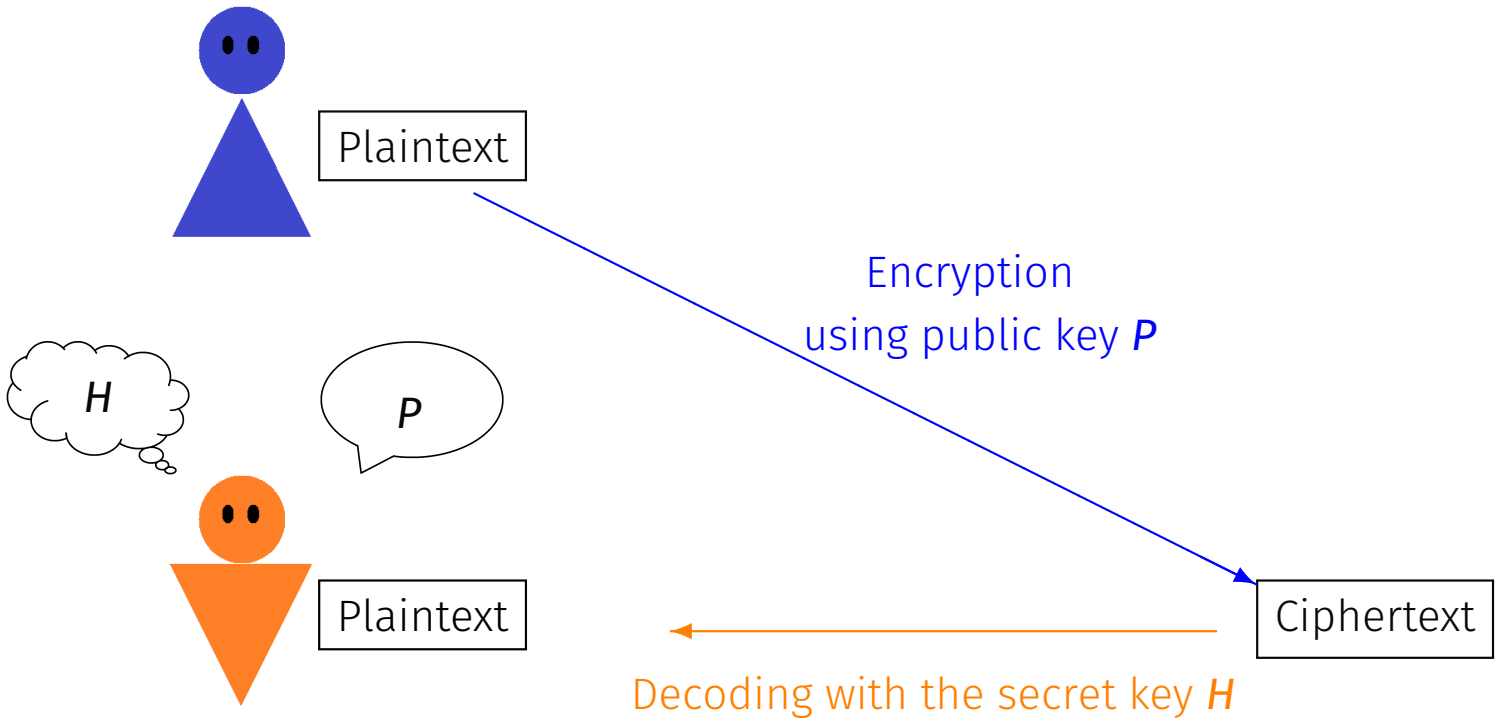


Secret key : a QC MDPC matrix $H = (H_0, H_1)$
Public key : a matrix $P = H_1^{-1} H_0$

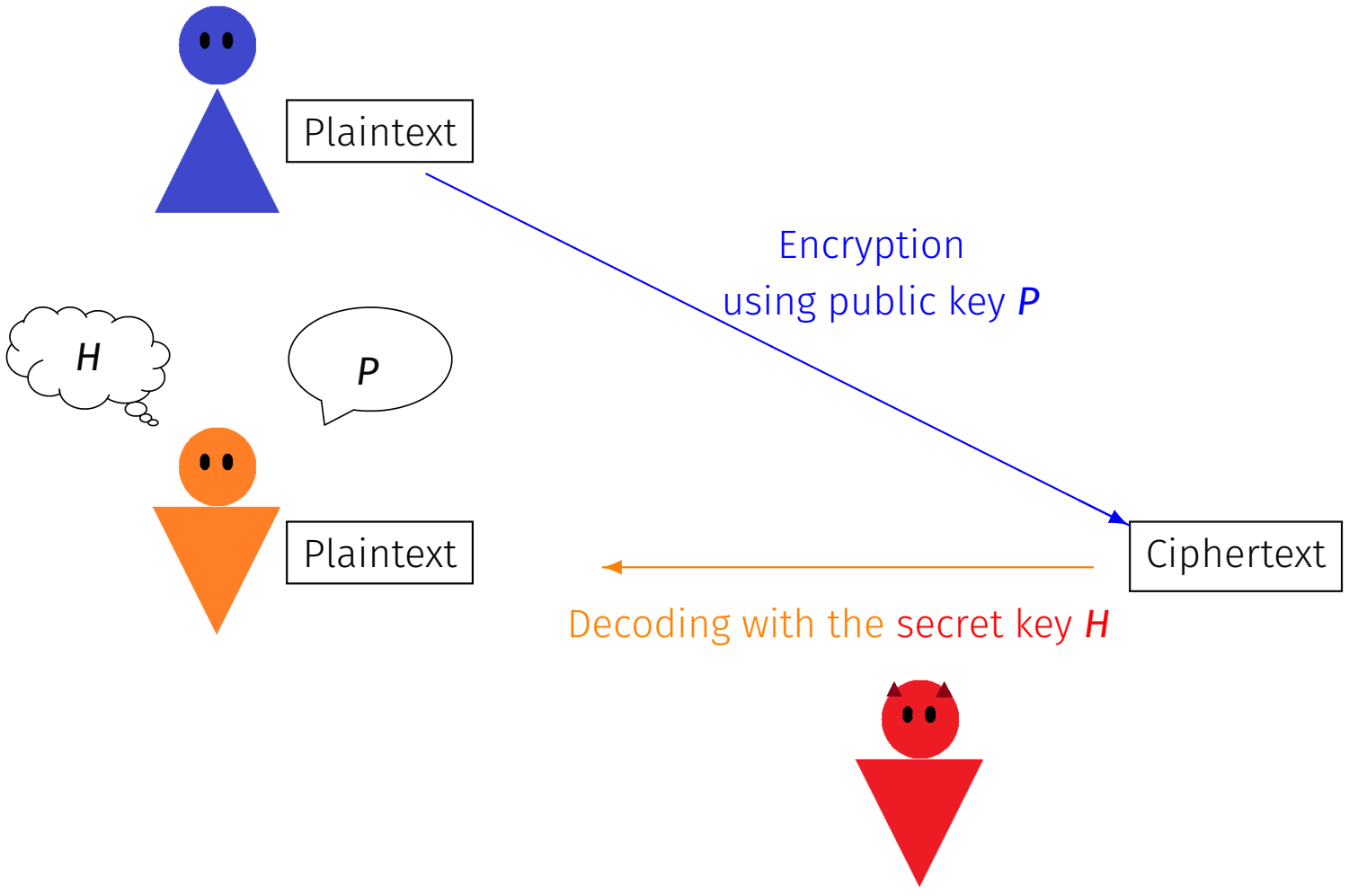


- We want to know the secret key H
- We know the public key P
- We know some ciphertexts previously sent
- We have access to the power traces

Secret key : a QC MDPC matrix $H = (H_0, H_1)$
Public key : a matrix $P = H_1^{-1} H_0$



Secret key : a QC MDPC matrix $H = (H_0, H_1)$
Public key : a matrix $P = H_1^{-1} H_0$



Bit Flipping

Algorithm 1: Bit Flipping

Data: $H \in \mathbb{F}_2^{r \cdot n}, x \in \mathbb{F}_2^n$
Result: Corrected codeword v

1 $v \leftarrow x$;
2 $S \leftarrow H \cdot v^T$ // Syndrome computation
3 ...
4 Computation of the error e
5 ...
6 Return the codeword $v = x \oplus e$

Our contribution

- New classical key recovery attack
  1. Differential Power Analysis (DPA)
  2. Mathematical key recovery

$$\boldsymbol{H} = (H_0, H_1)$$

- New classical key recovery attack
  1. Differential Power Analysis (DPA)
  2. Mathematical key recovery

$$H = \left(\begin{pmatrix} * & \cdots & * \\ \vdots & & \vdots \\ * & \cdots & * \end{pmatrix} \quad \begin{pmatrix} * & \cdots & * \\ \vdots & & \vdots \\ * & \cdots & * \end{pmatrix} \right)$$

- New classical key recovery attack
  1. Differential Power Analysis (DPA)
  2. Mathematical key recovery

$$H = \left(\begin{pmatrix} 0 & * & * & 0 & * & * \\ * & & \cdots & & & * \\ \vdots & & & & & \vdots \\ * & & \cdots & & & * \end{pmatrix} \quad \begin{pmatrix} * & \cdots & * \\ \vdots & & \vdots \\ * & \cdots & * \end{pmatrix} \right)$$

- New classical key recovery attack
  1. Differential Power Analysis (DPA)
  2. Mathematical key recovery

$$H = \left(\begin{pmatrix} 0 & * & * & 0 & * & * \\ * & 0 & * & * & 0 & * \\ * & * & 0 & * & * & 0 \\ 0 & * & * & 0 & * & * \\ * & 0 & * & * & 0 & * \\ * & * & 0 & * & * & 0 \end{pmatrix} \quad \begin{pmatrix} * & \cdots & * \\ \vdots & & \vdots \\ * & \cdots & * \end{pmatrix} \right)$$

- New classical key recovery attack
  1. Differential Power Analysis (DPA)
  2. Mathematical key recovery

$$H = \left(\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} * & \cdots & * \\ \vdots & & \vdots \\ * & \cdots & * \end{pmatrix} \right)$$

- New classical key recovery attack
  1. Differential Power Analysis (DPA)
  2. Mathematical key recovery

$$H = \left(\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \quad \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \right)$$

Our contribution

- New classical key recovery attack
  1. Differential Power Analysis (DPA)
  2. Mathematical key recovery
- Our countermeasure

DIFFERENTIAL POWER ANALYSIS

Syndrome calculation inside the Bit Flipping

$$H \cdot \begin{pmatrix} c^T \\ 0 \end{pmatrix} = (H_0, H_1) \cdot \begin{pmatrix} c^T \\ 0 \end{pmatrix} = H_0 \cdot c^T$$

H_0 is a sparse circulant matrix.
H_0 is uniquely defined by $\{x_0, ..., x_{44}\}$, the unknown indices of the nonzero elements of its first row.

Recovering the $\{x_0, ..., x_{44}\}$ means recovering the whole matrix H_0.

$$H_0 = \begin{array}{c} \begin{matrix} & & x_0 & & & x_1 \\ & & \downarrow & & & \downarrow \end{matrix} \\ \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array} \quad \} \quad h_0$$

$$H_0 \cdot c^T$$

During the multiplication, H_0 is decomposed as a sum of 45 rotation matrices

$$H_0 = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

$$H_0 \cdot c^T$$

The multiplication algorithm runs through all the rotations composing H_0 and computes the intermediate rotated ciphertexts $r_{x_i}(c)^T$

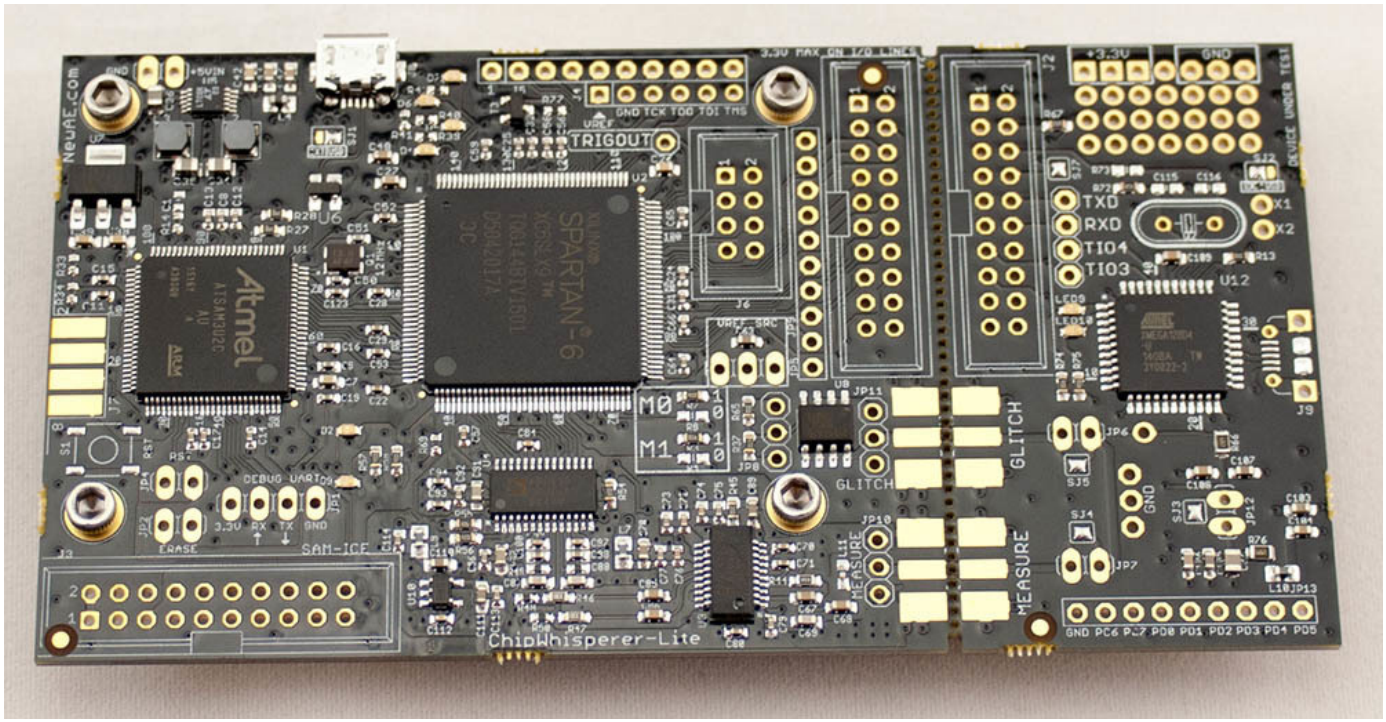
$$H_0 \cdot c^T = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \cdot c^T + \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \cdot c^T$$

$$H_0 \cdot c^T = r_{x_0}(c)^T + r_{x_1}(c)^T$$

The final value of the multiplication is the xor of all the $r_{x_i}(c)^T$

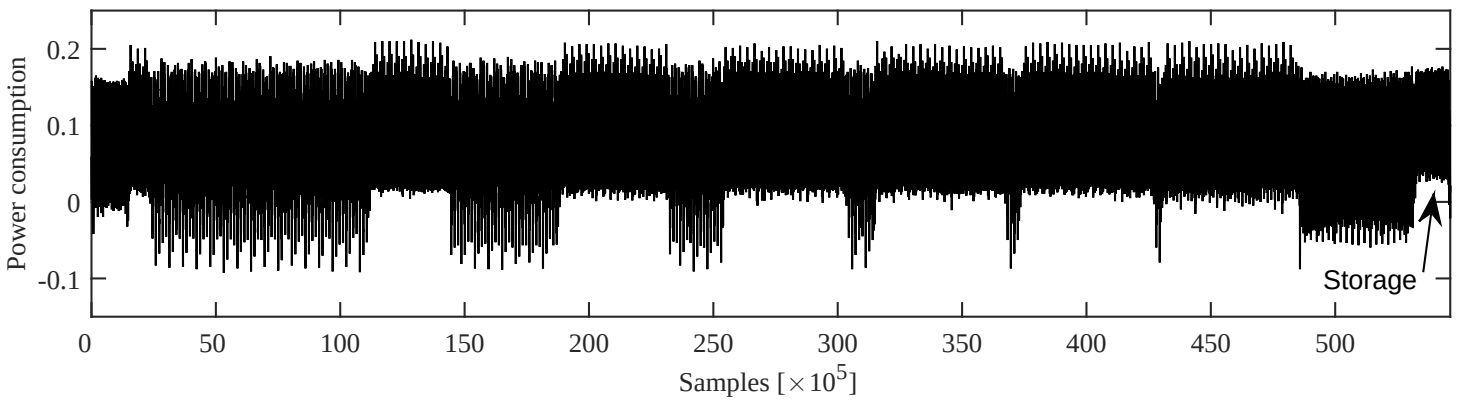
ChipWhisperer Lite

- Original code
- Programmable chip (Atmel AVR XMEGA128)
- Easy to use : On-board power-measurement circuit
- Easily reproducible



Target : The storing into local memory of each $\boldsymbol{r}_{x_i}(\boldsymbol{c})^T$

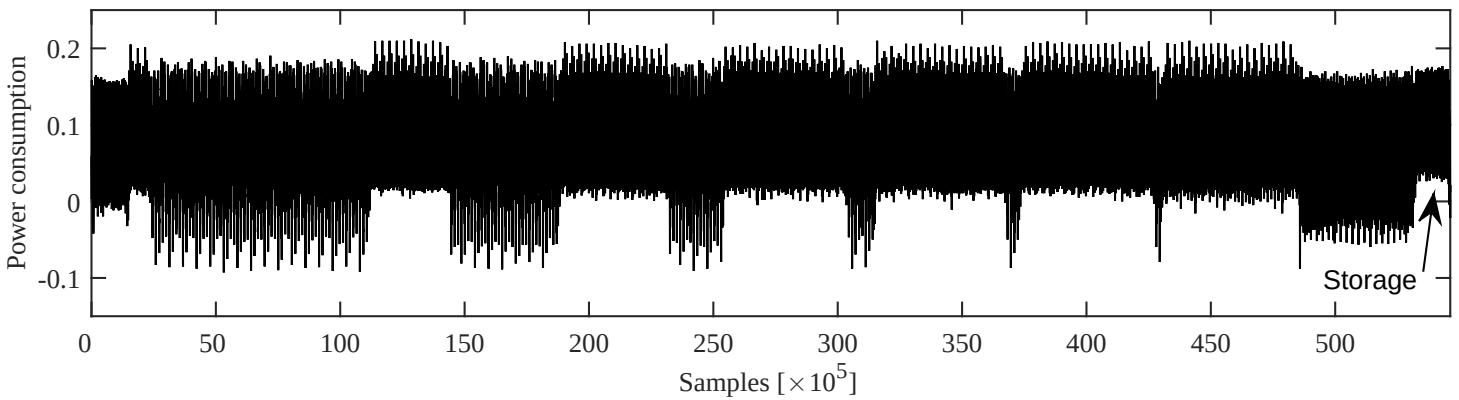
Power trace of a rotation computation



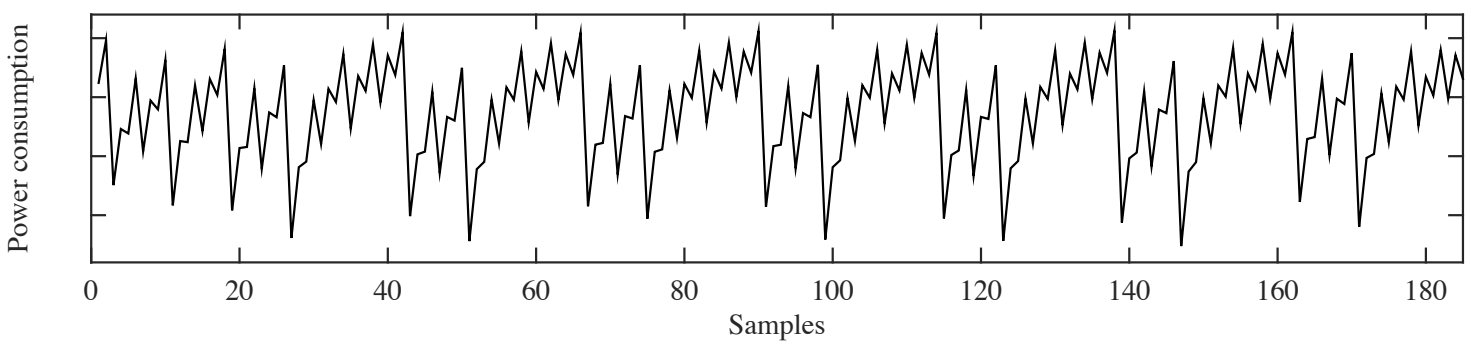
Zoom on the storage

Target : The storing into local memory of each $\boldsymbol{r}_{x_i}(\boldsymbol{c})^T$

Power trace of a rotation computation

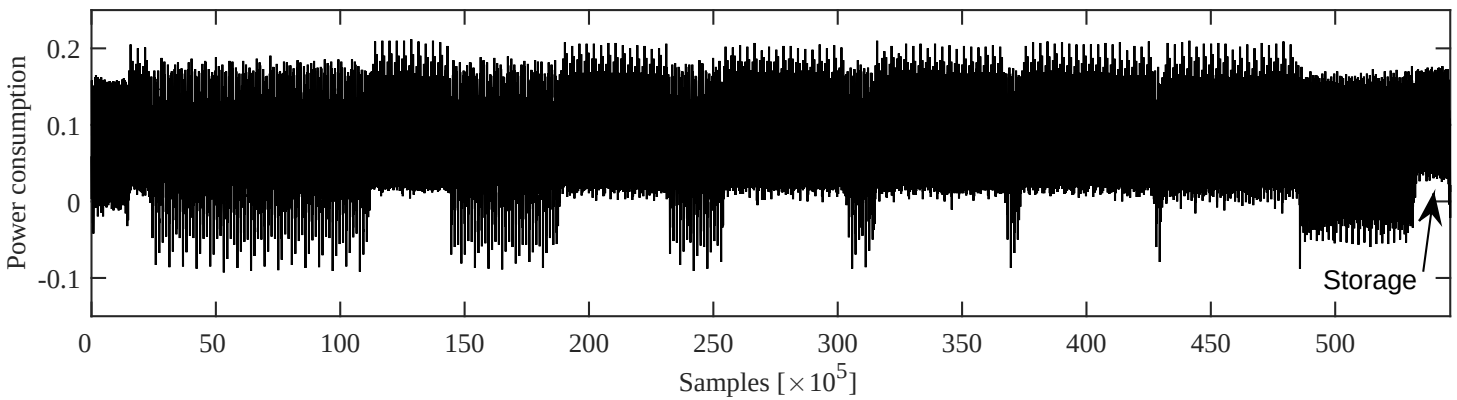


Zoom on the storage

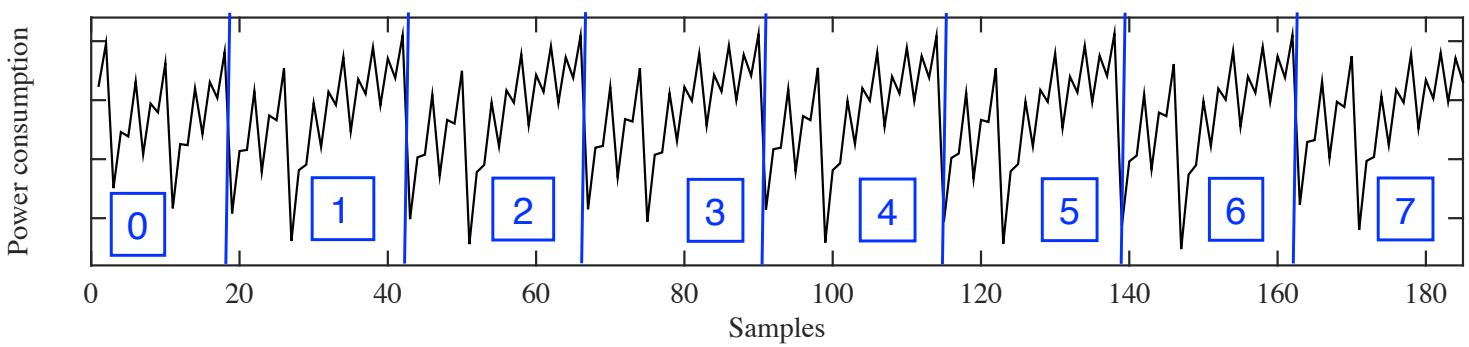


Target : The storing into local memory of each $\boldsymbol{r}_{x_i}(\boldsymbol{c})^T$

Power trace of a rotation computation

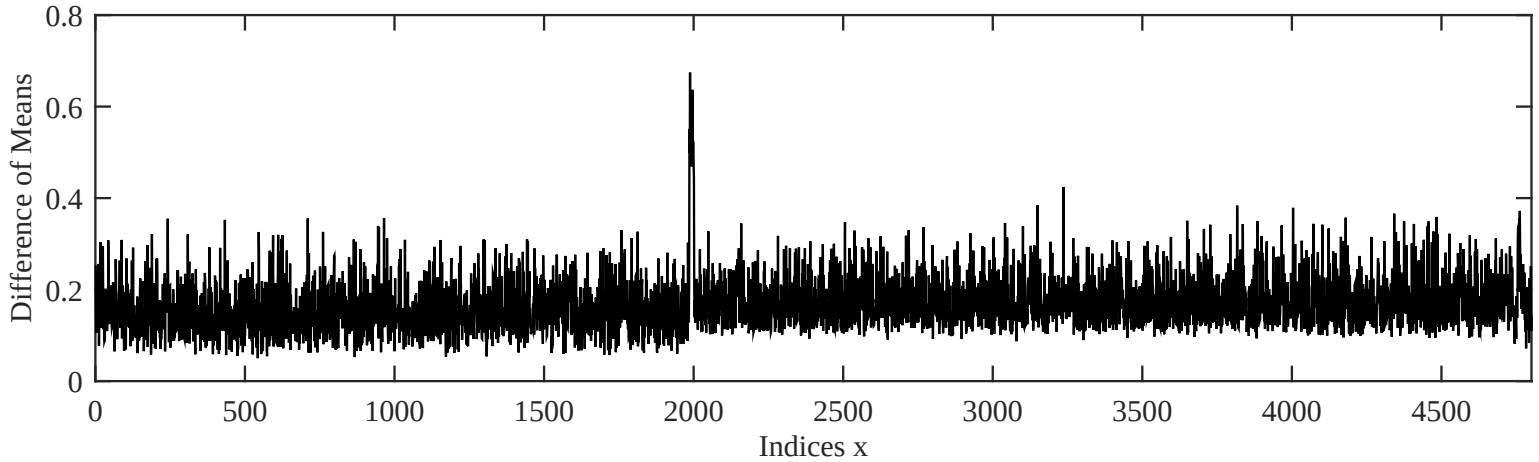


Zoom on the storage



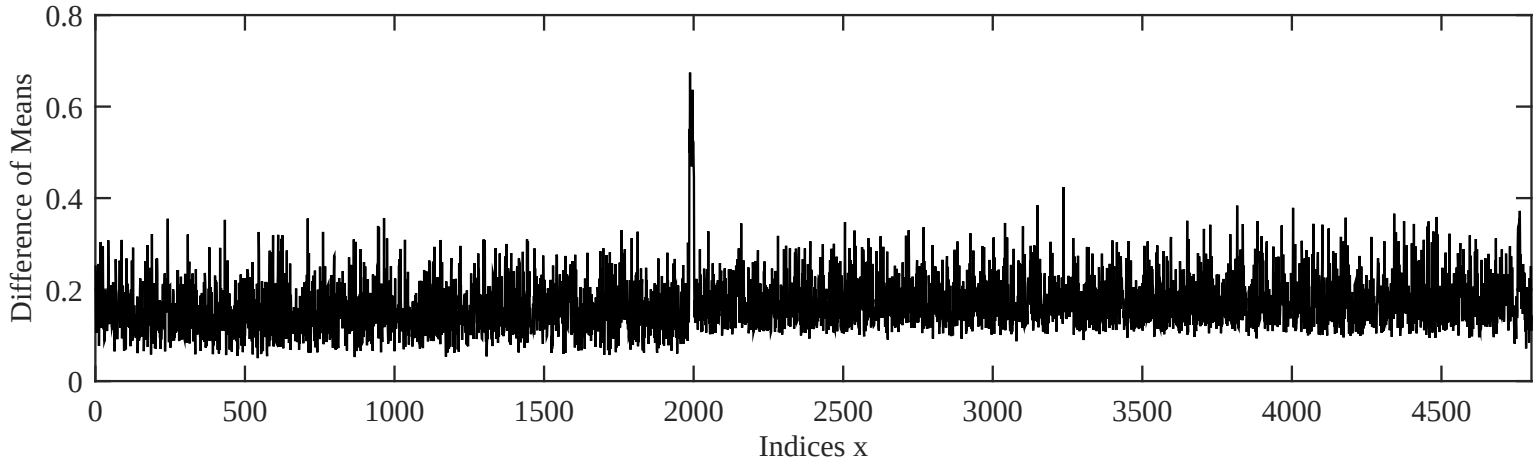
DPA Results

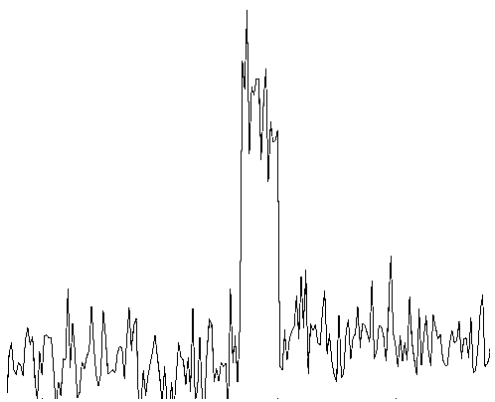
Maximum Difference of Means (DoM) using 500 traces over all possible values.
Significant difference is observed around the correct index $x_i = 2000$.



DPA Results

Maximum Difference of Means (DoM) using 500 traces over all possible values.
Significant difference is observed around the correct index $x_i = 2000$.

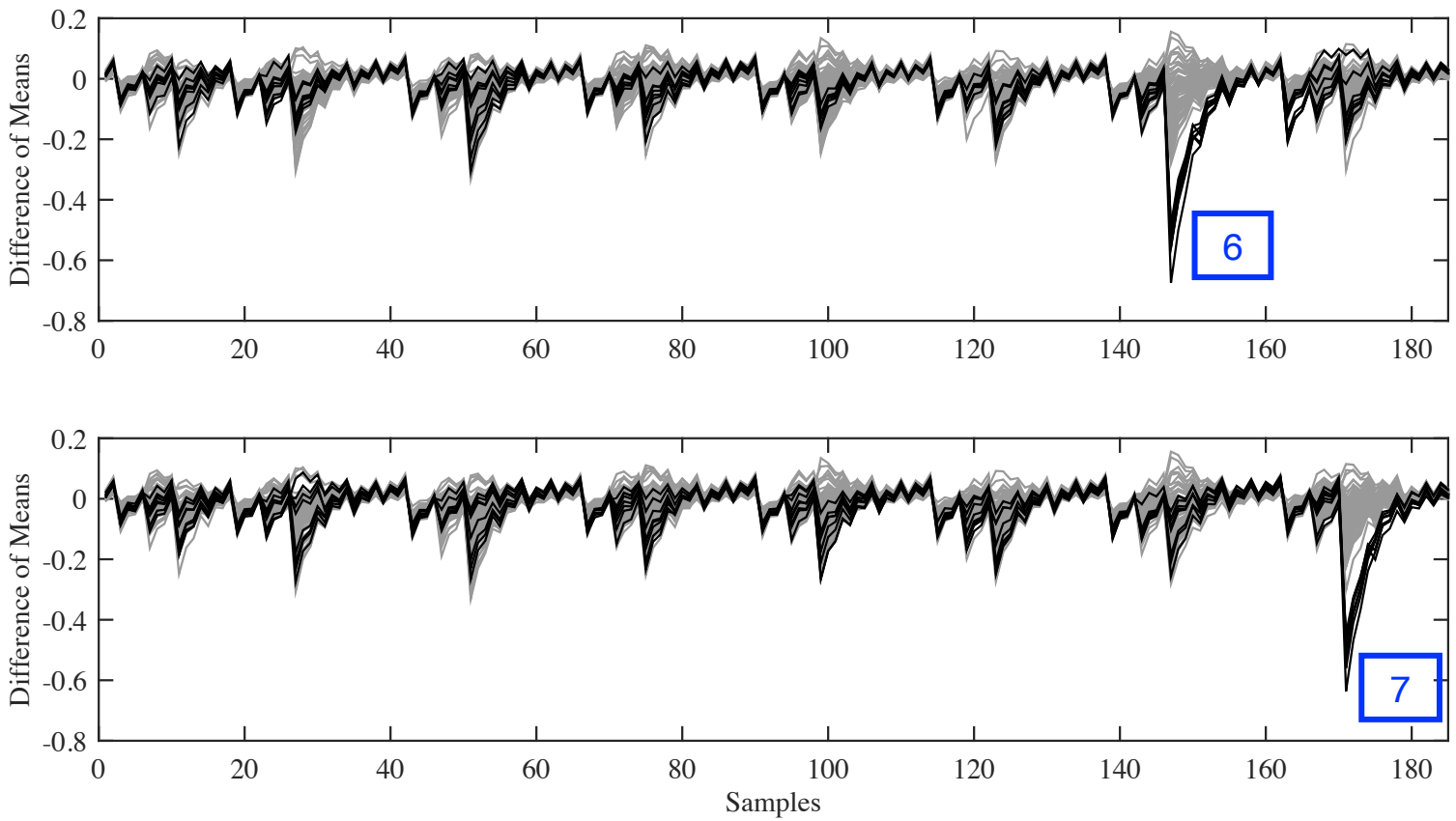




The peak always starts on a multiple of 64.

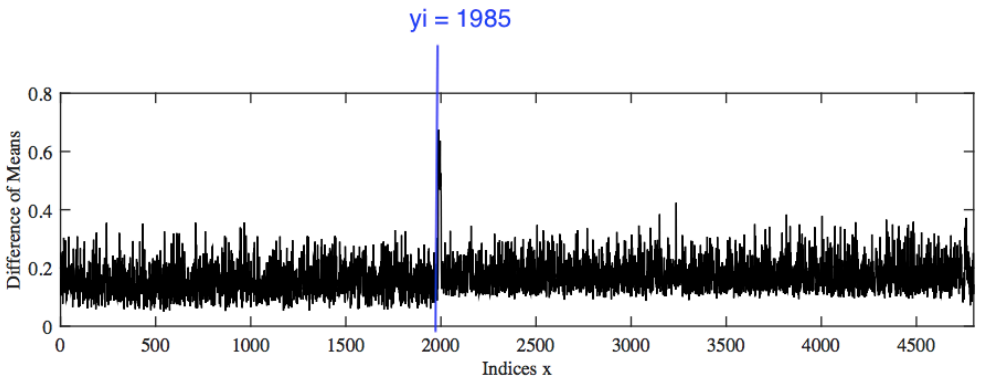
DPA Results

Let's look at the leak in time.



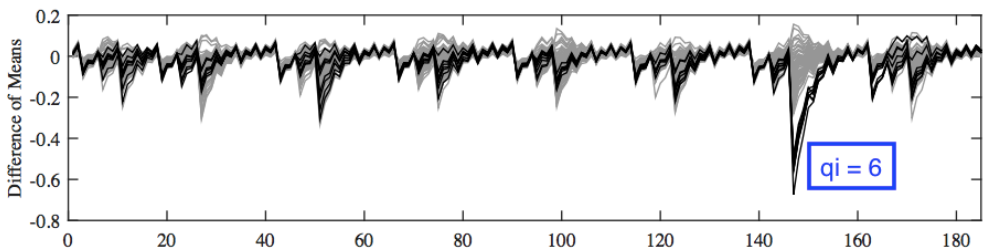
DPA Results

Leakage model 1



$$y_i = \lfloor \frac{(x_i - 1) \ mod \ r}{64} \rfloor \cdot 64 + 1$$

Leakage model 2



$$q_i = 7 - \lfloor \frac{(x_i - 1) \bmod 64}{8} \rfloor$$

DPA Results

If we combine leakage models 1 and 2
$\rightarrow$ only 8 possible values for x_i

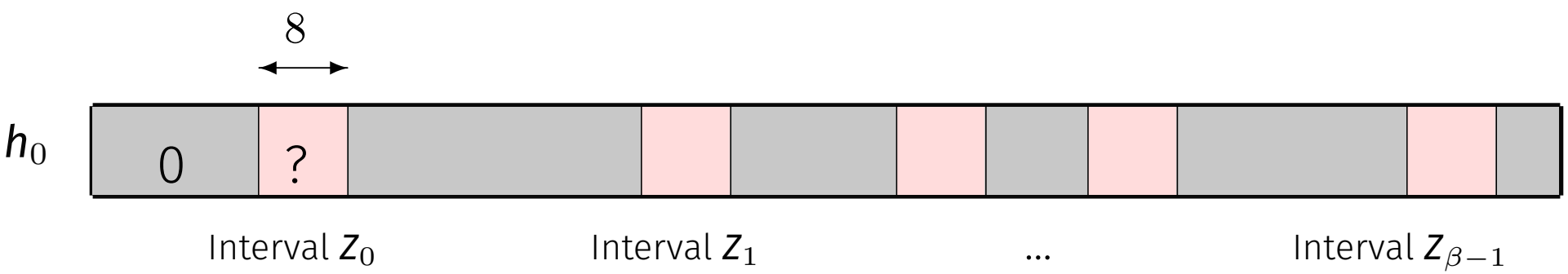
$$x_i \in Z_i = [y_i + 8(7 - q_i),\ y_i + 8(7 - q_i) + 7]$$

In our example, we measured $(y_i, q_i) = (1985, 6)$
and therefore deduce that $Z_i = [1993, 2000]$.

If we combine leakage models 1 and 2
$\rightarrow$ only 8 possible values for x_i

$$x_i \in Z_i = [y_i + 8(7 - q_i),\ y_i + 8(7 - q_i) + 7]$$

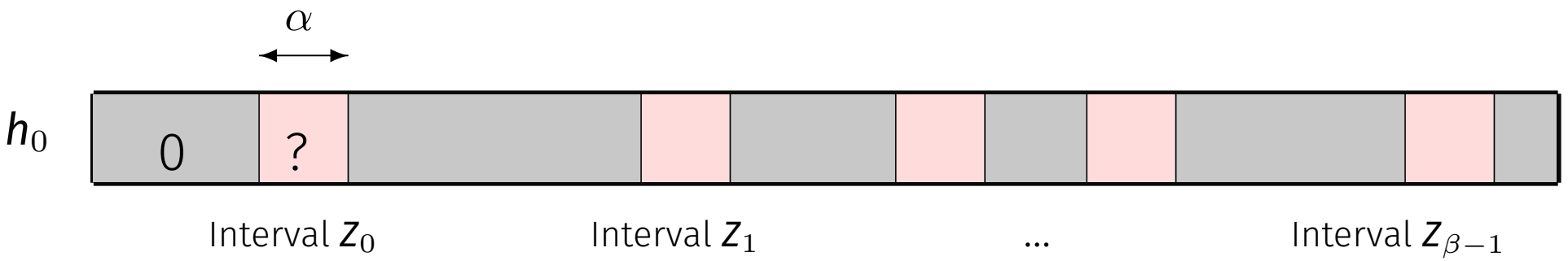
In our example, we measured $(y_i, q_i) = (1985, 6)$
and therefore deduce that $Z_i = [1993, 2000]$.



Knowledge of H_0 after DPA attack

$\alpha \leftarrow$ length of index search intervals Z_i.

$\beta \leftarrow$ total number of unique search intervals Z_i.



α represents DPA attack accuracy

KEY RECOVERY

Setting up a system of noisy binary linear equations

Recall that the public key is $P = H_1^{-1} \cdot H_0$.

Setting $Q = P^{-T}$ we rearrange and write

$$Q \cdot h_0^T = h_1^T$$

where

- Q is dense and known
- h_0 (the first row of H_0) is sparse and partially known
- h_1 (the first row of H_1) is sparse and unknown.

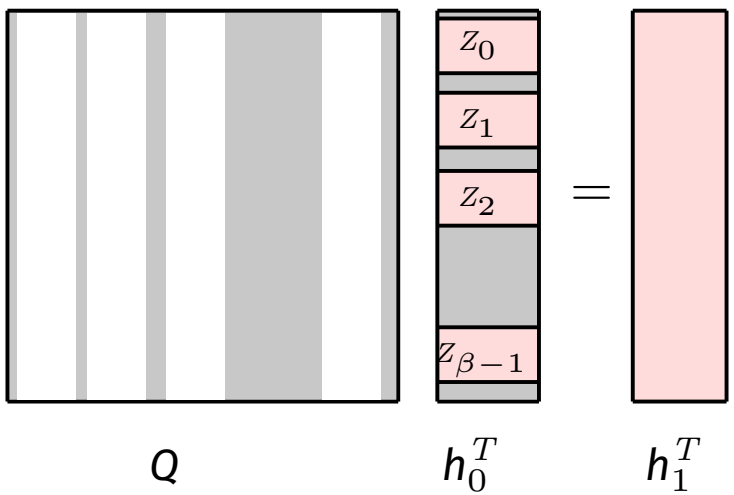
Setting up a system of noisy binary linear equations

Recall that the public key is $P = H_1^{-1} \cdot H_0$.

Setting $Q = P^{-T}$ we rearrange and write

$$Q \cdot h_0^T = h_1^T$$

STEP 1 : Remove columns of Q



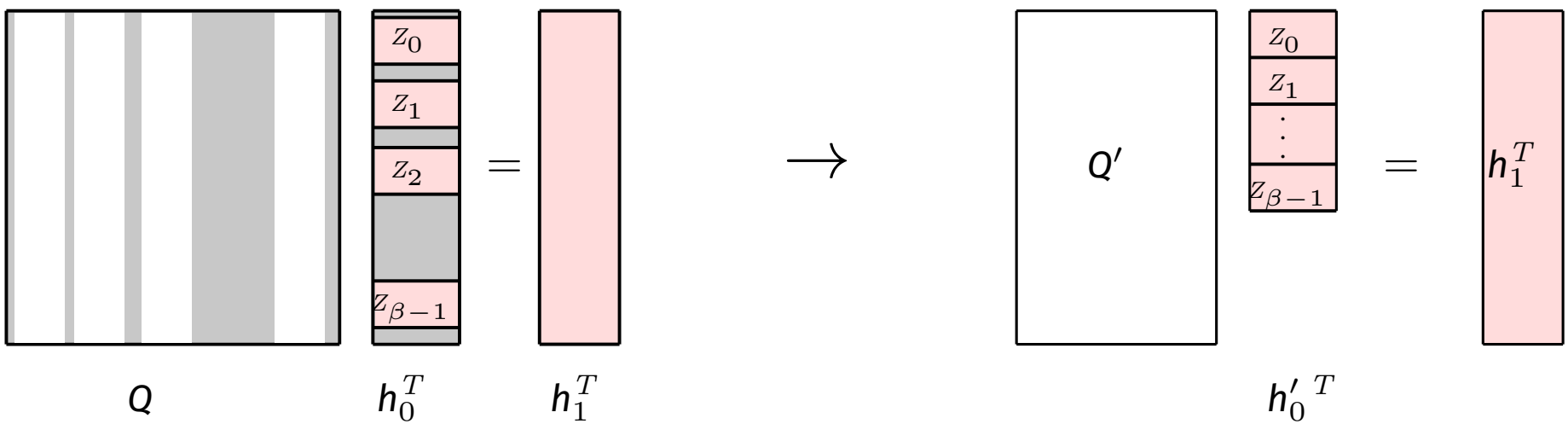
Setting up a system of noisy binary linear equations

Recall that the public key is $P = H_1^{-1} \cdot H_0$.

Setting $Q = P^{-T}$ we rearrange and write

$$Q \cdot h_0^T = h_1^T$$

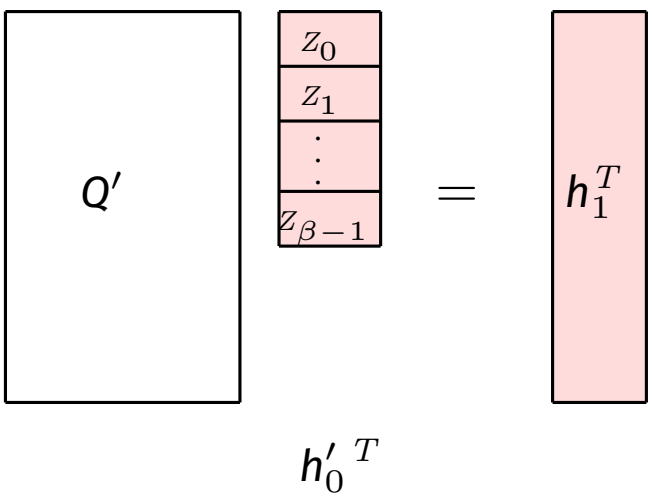
STEP 1 : Remove columns of Q



Setting up a system of noisy binary linear equations

STEP 2 : Add parity equations

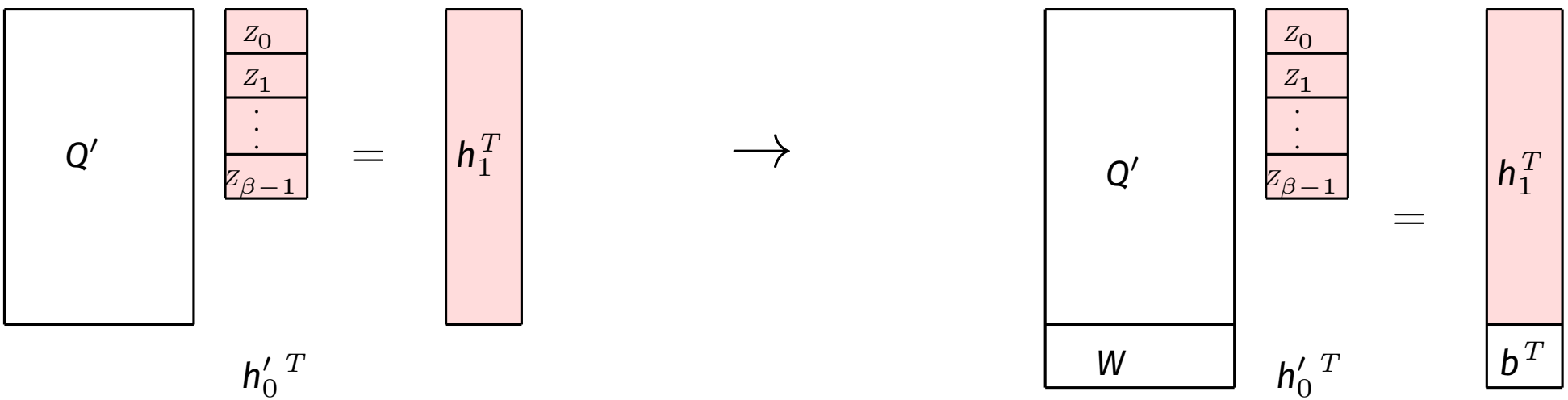
DPA $\rightarrow$ number of nonzero values of each interval Z_i of h_0



Setting up a system of noisy binary linear equations

STEP 2 : Add parity equations

DPA $\rightarrow$ number of nonzero values of each interval Z_i of h_0



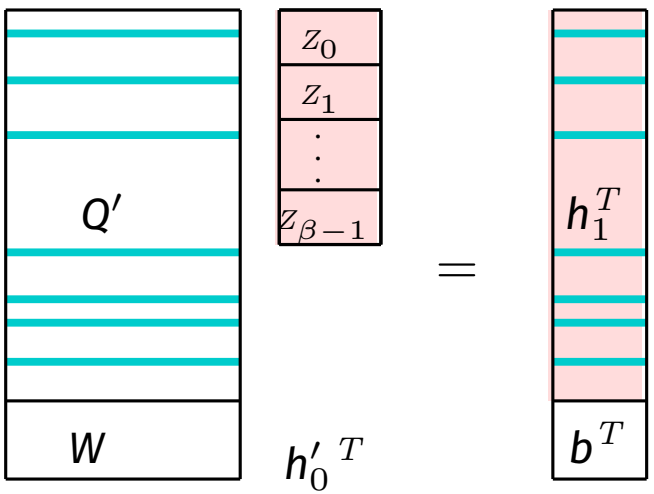
STEP 2 : Guess some zeros of h_1

h_1 is an extremely sparse vector. Its entries are zero with probability $1 - \frac{w}{2r} > 0.99$

Setting up a system of noisy binary linear equations

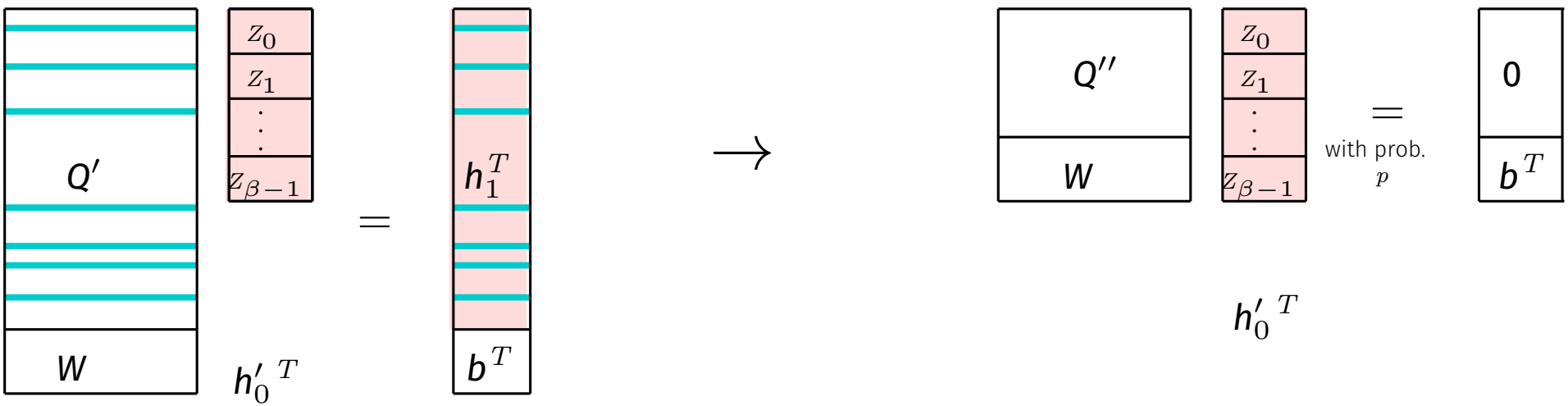
STEP 2 : Guess some zeros of h_1

We create a square system of equations by randomly selecting entries from h_1, and keeping the corresponding rows of Q'.



Setting up a system of noisy binary linear equations

STEP 2 : Guess some zeros of h_1



Complexity

Average number of attempts ($= \frac{1}{p}$) before getting a correct system

DPA accuracy (α)	8	16	32	64
80-bit	22	950	2^{23}	2^{58}
128-bit	40	3500	2^{26}	2^{64}

Complexity

Average number of attempts ($= \frac{1}{p}$) before getting a correct system

DPA accuracy (α)	8	16	32	64
80-bit	22	950	2^{23}	2^{58}
128-bit	40	3500	2^{26}	2^{64}

Total complexity in terms of multiplications in $\mathbb{F}_2$

$$\frac{1}{p} \cdot \left(\frac{w\alpha}{2}\right)^{2.8}$$

r	w	Bits of Security
4801	90	80
9857	142	128

Average number of attempts ($= \frac{1}{p}$) before getting a correct system

DPA accuracy (α)	8	16	32	64
80-bit	22	950	2^{23}	2^{58}
128-bit	40	3500	2^{26}	2^{64}

Total complexity in terms of multiplications in $\mathbb{F}_2$

$$\frac{1}{p} \cdot \left(\frac{w\alpha}{2}\right)^{2.8}$$

r	w	Bits of Security
4801	90	80
9857	142	128

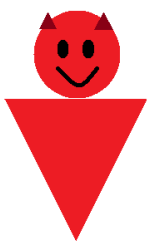
In our device ($\alpha = 8$), we have

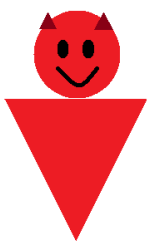
	80-bit	128-bit
Complexity	2^{28}	2^{31}

Experimental results

SAGE on one core of a 2.9GHz Core i5 MacBook Pro [1]

DPA accuracy (α)	8	16	32	64
80-bit	0.4 sec	15 sec	16 hours	$\geq$ 600 years
128-bit	2 sec	4 min	7 days	$\geq$ 800,000 years



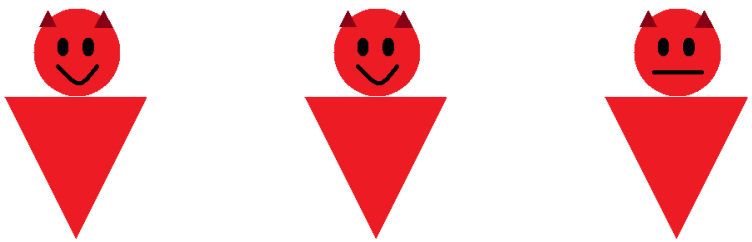


[1] https://www.di.ens.fr/~mrossi/

Experimental results

SAGE on one core of a 2.9GHz Core i5 MacBook Pro [1]

DPA accuracy (α)	8	16	32	64
80-bit	0.4 sec	15 sec	16 hours	$\geq$ 600 years
128-bit	2 sec	4 min	7 days	$\geq$ 800,000 years

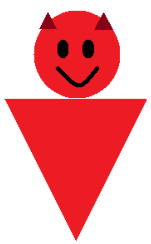
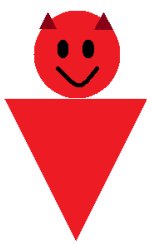
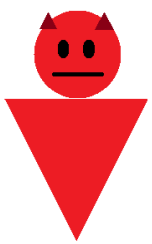
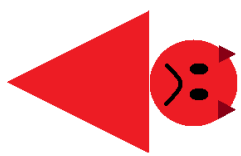


[1] https://www.di.ens.fr/~mrossi/

Experimental results

SAGE on one core of a 2.9GHz Core i5 MacBook Pro [1]

DPA accuracy (α)	8	16	32	64
80-bit	0.4 sec	15 sec	16 hours	$\geq$ 600 years
128-bit	2 sec	4 min	7 days	$\geq$ 800,000 years

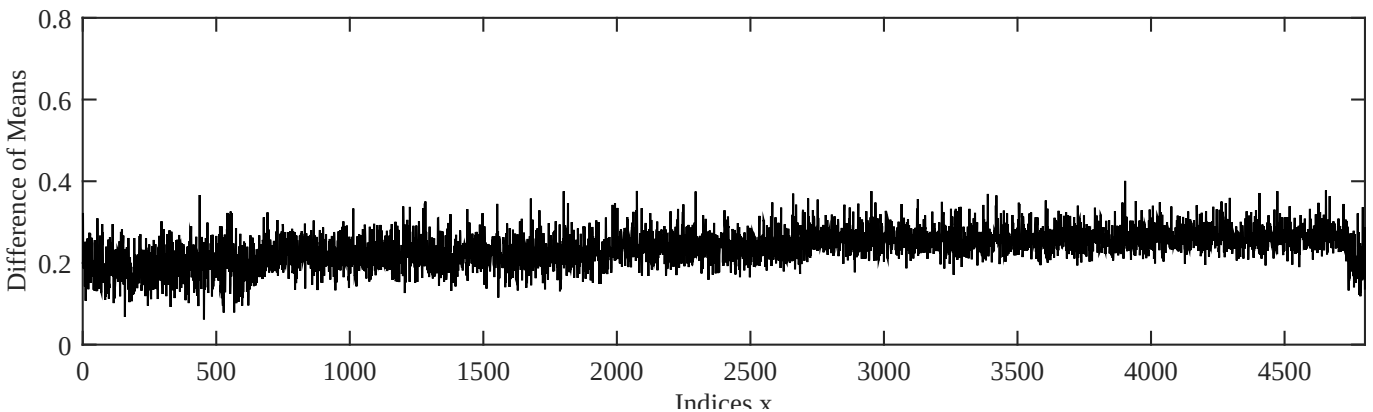
[1] https://www.di.ens.fr/~mrossi/

COUNTERMEASURE

Simple Masking Countermeasure

→ Let's mask the corrupted codeword $(\boldsymbol{c} \mid \boldsymbol{0})$ by XORing it with a random codeword $\boldsymbol{c}_m$

$$H \cdot \left(\left(\boldsymbol{c}|\boldsymbol{0}\right) \oplus \boldsymbol{c}_m \right)^T = H \cdot \left(\boldsymbol{c} \mid \boldsymbol{0}\right)^T \oplus H \cdot \boldsymbol{c}_m{}^T = H \cdot \left(\boldsymbol{c} \mid \boldsymbol{0}\right)^T$$



Maximum of the Difference Of Means with the countermeasure enabled (500 traces)

Final Analysis of QcBits

QcBits

Advantages	Drawbacks
· Post Quantum candidate · Small key sizes · Very efficient · Quite easy to protect against DPA	· Sparseness of the secret keys can be a weakness · Non negligible failure rate $\Rightarrow$ Attack in the non ephemeral case Guo et al (Asiacrypt 2016)

Final Analysis of QcBits

QcBits

Advantages	Drawbacks
· Post Quantum candidate · Small key sizes · Very efficient · Quite easy to protect against DPA	· Sparseness of the secret keys can be a weakness · Non negligible failure rate $\Rightarrow$ Attack in the non ephemeral case Guo et al (Asiacrypt 2016)

Thank you for your attention !