Side-Channel Attacks against HQC and Countermeasures

Guillaume GOY

XLIM, University of Limoges

18 March 2025

Modern cryptography

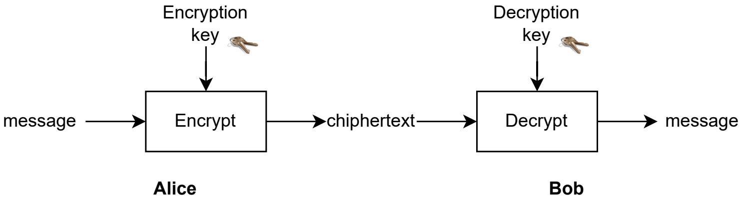


Figure – Overview of a cryptosystem

Hybrid Cryptosystem :

- Symmetric-key cryptography : based on exhaustive key research
- Public-key cryptography : based on a hard problem

→ RSA [RSA78] - Elliptic Curves Cryptography (ECC) [Kob87, Mil85]

Post-Quantum Cryptography (PQC) – NIST Standardization

Figure – IBM Quantum Computer

→ Quantum Computer threat !
Shor's and Grover's Algorithms

Post-Quantum Cryptography (PQC) – NIST Standardization

Figure – IBM Quantum Computer

→ Quantum Computer threat !

Shor's and Grover's Algorithms

Several possibilities (NIST Standards) :

- Kyber (ML-KEM - FIPS203) [BDK+18]
- Dilithium (ML-DSA - FIPS024) [DKL+18]
- Falcon (not yet published) [PFH+20]
- Sphincs+ (SLH-DSA - FIPS205) [BHK+19]
- HQC (not yet published) [AMAB+17]

Other past code-based candidates :

- BIKE [ABB+17] // ClassicMcEliece [BCL+]

And now ? ! → new round for additionnal signature schemes !
(promizing MPC-in-the-head ? !)

Cryptographic Security

We consider three levels of security : (I) 2^{128}, (III) 2^{192} and (IV) 2^{256}

This represents the **minimal number of operation requiered to recover a secret information**.

And often also **The number of different secret keys**.

Cryptographic Security

We consider three levels of security : (I) 2^{128}, (III) 2^{192} and (IV) 2^{256}

This represents the **minimal number of operation requiered to recover a secret information**.

And often also **The number of different secret keys**.

$$2^{128} = \underbrace{2^{33}}_{\substack{\text{8.6 billion}\\ \text{Number of}\\ \text{human beings}\\ \text{on earth}}} \times \underbrace{2^{33}}_{\substack{\text{8.6 GHz}\\ \text{CPU frequency}}} \times \underbrace{2^{62}}$$

Cryptographic Security

We consider three levels of security : (I) 2^{128}, (III) 2^{192} and (IV) 2^{256}

This represents the **minimal number of operation requiered to recover a secret information**.

And often also **The number of different secret keys**.

$$2^{128} = \underbrace{2^{33}}_{\substack{8.6 \text{ billion} \\ \text{Number of} \\ \text{human beings} \\ \text{on earth}}} \times \underbrace{2^{33}}_{\substack{8.6 \text{ GHz} \\ \text{CPU frequency}}} \times \underbrace{2^{62}}_{\substack{> 146 \text{ billion years} \\ > 10\times \text{ Age of the Universe}}}$$

Cryptographic Security

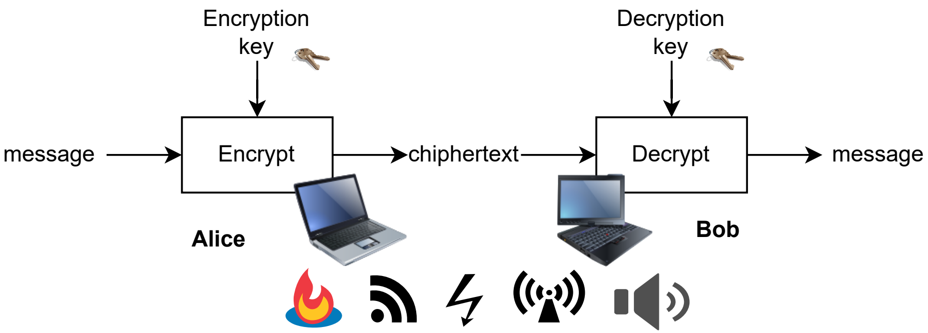
We consider three levels of security : (I) 2^{128}, (III) 2^{192} and (IV) 2^{256}
This represents the **minimal number of operation requiered to recover a secret information**.
And often also **The number of different secret keys**.

$$2^{128} = \underbrace{2^{33}}_{\substack{\text{8.6 billion}\\ \text{Number of}\\ \text{human beings}\\ \text{on earth}}} \times \underbrace{2^{33}}_{\substack{\text{8.6 GHz}\\ \text{CPU frequency}}} \times \underbrace{2^{62}}_{\substack{> \text{146 billion years}\\ > 10\times \text{ Age of the Universe}}}$$

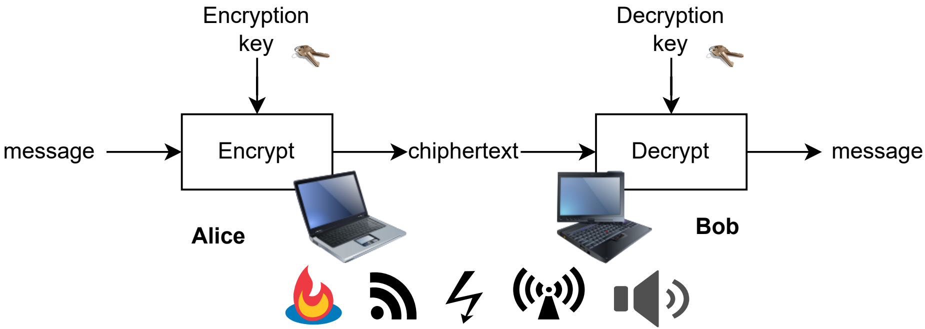
$$2^{256} \approx\approx 10^{80} \leftarrow \text{Number of atoms in the observable universe}$$

Number of worldwide operations for Bitcoin in a year $\approx 2^{95}$.

Side-Channel Attacks



Side-Channel Attacks



Physical behavior is correlated to manipulated data.
The first side-channel attack was introduced by Paul Kocher in 1996 [Koc96].

Side-channel attacks toy example

Side-channel attacks toy example

Random Digicode : 10^4 combinations

Side-channel attacks toy example

Random Digicode : 10^4 combinations
Worn Digicode : 24 combinations

- Bypass the security with a physical observation

Table of Contents

Table of Contents

Error Correcting Codes

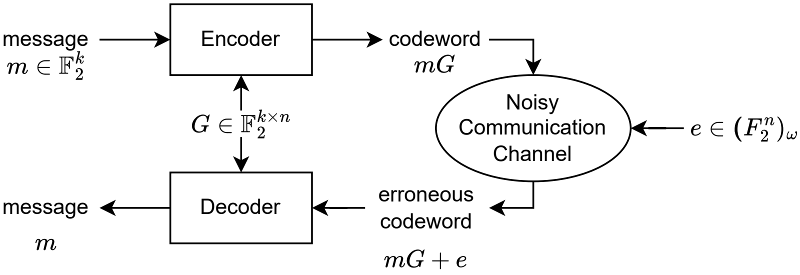


Figure – Overview of an Error Correcting Code.

Code-based cryptography : $G \xleftarrow{\$} \mathbb{F}_2^{k \times n}$, $m \xleftarrow{\$} \mathbb{F}_2^k$ and $e \xleftarrow{\$} (\mathbb{F}_2^n)_\omega$.

Decoding Problem :

Given $(mG + e, G)$, it is hard to recover m (NP-complete [BMVT78]).

Building Code-based cryptography

(i) Mask the Code with a random permutation [McE78][ABB+17]

Building Code-based cryptography

(i) Mask the Code with a random permutation [McE78][ABB+17]

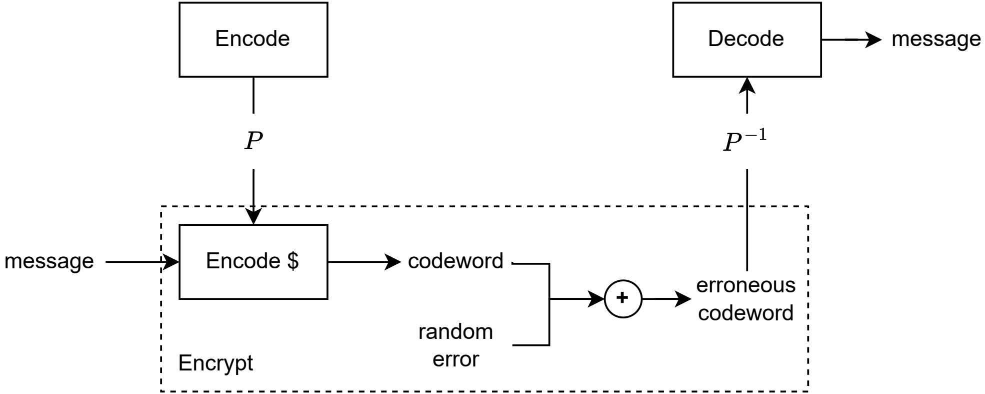


Figure – Masking error correcting code structure to build cryptography

Building Code-based cryptography

(i) Mask the Code with a random permutation [McE78][ABB$^+$17]

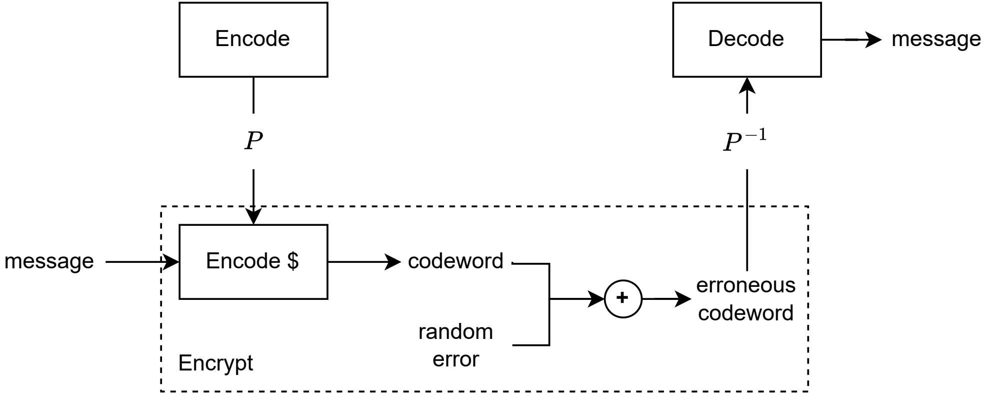


Figure – Masking error correcting code structure to build cryptography

Hamming Quasi-Cyclic (HQC)

$$\mathcal{R} = \mathbb{F}_2[X]/(X^n - 1)$$

Alice		Bob
$\mathbf{h} \xleftarrow{\$} \mathcal{R}$ $\quad \mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{R}_\omega^2$ $\mathbf{s} = \mathbf{x} + \mathbf{h}\mathbf{y}$	$\xrightarrow{\mathbf{s}, \mathbf{h}}$	message m $\mathbf{e} \xleftarrow{\$} \mathcal{R}_{\omega_r}$ $\quad \mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{R}_{\omega_e}^2$ $\mathbf{u} = \mathbf{r}_1 + \mathbf{h}\mathbf{r}_2$
$\mathbf{m}' = \mathcal{C}.\mathtt{Decode}(\mathbf{v} - \mathbf{u}\mathbf{y})$	$\xleftarrow{\mathbf{u}, \mathbf{v}}$	$\mathbf{v} = \mathcal{C}.\mathtt{Encode}(\mathbf{m}) + \mathbf{s}\mathbf{r}_2 + \mathbf{e}$

Figure – HQC Public Key Encryption Scheme

- No Code structure masking

2 codes for HQC :

- $\mathbf{h}$ is a random code to protect the secret key and perform the encryption.
- $\mathcal{C}$ is a public and efficient code to perform decryption. Any code can be selected.

Hamming Quasi-Cyclic (HQC) 2

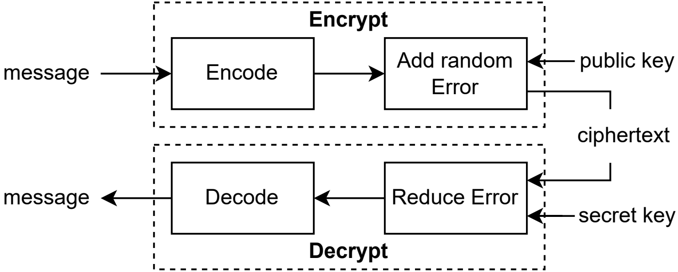


Figure – Hamming Quasi-Cyclic Overview

Concatenated Code structure

- Before 2019 → Concatenated BCH and repetition codes.
- After 2019 → Concatenated Reed-Muller and Reed-Solomon codes.

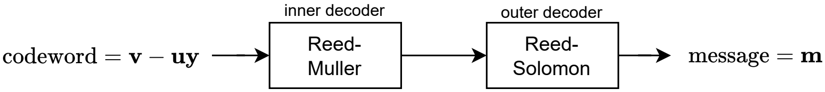


Figure – HQC Concatenated codes structure

Concatenated Code structure

- Before 2019 $\rightarrow$ Concatenated BCH and repetition codes.
- After 2019 $\rightarrow$ Concatenated Reed-Muller and Reed-Solomon codes.

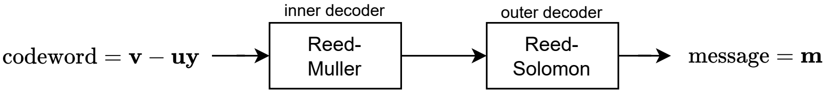


Figure – HQC Concatenated codes structure

(i) **Secret key** recovery attacks : [SHR+22, GLG22a, BMG+24]

(ii) **Shared key** (message) recovery attacks : [GLG22b, GMGL23, BMG+24]

Table of Contents

Attack Scenario I

→ Chosen Ciphertext attack to recover the secret key $\mathbf{y}$.

$$\mathcal{C}.\texttt{Decode}(\mathbf{v} - \mathbf{uy})$$

Attack Scenario I

$\rightarrow$ Chosen Ciphertext attack to recover the secret key $\mathbf{y}$.

$$\mathcal{C}.\texttt{Decode}(\mathbf{v} - \mathbf{u}\mathbf{y})$$

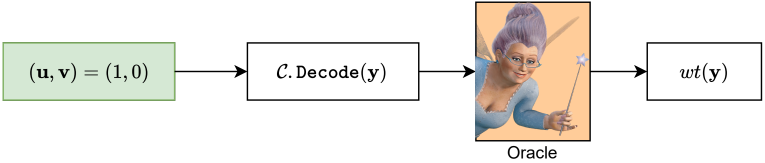
Choosing $\rightarrow (\mathbf{u}, \mathbf{v}) = (1, 0)$ leads to compute $\mathcal{C}.\texttt{Decode}(\mathbf{y})$

Attack Scenario I

→ Chosen Ciphertext attack to recover the secret key $\mathbf{y}$.

$$\mathcal{C}.\mathtt{Decode}(\mathbf{v} - \mathbf{u}\mathbf{y})$$

Choosing → $(\mathbf{u}, \mathbf{v}) = (1, 0)$ leads to compute $\mathcal{C}.\mathtt{Decode}(\mathbf{y})$

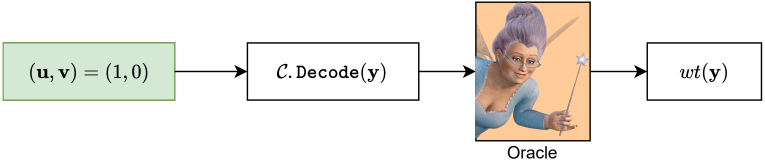


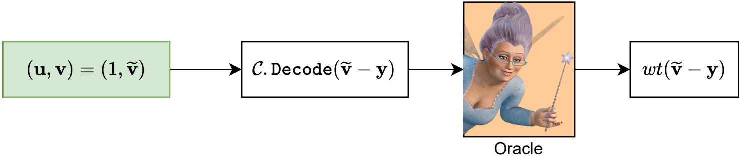
Attack Scenario I

→ Chosen Ciphertext attack to recover the secret key $\mathbf{y}$.

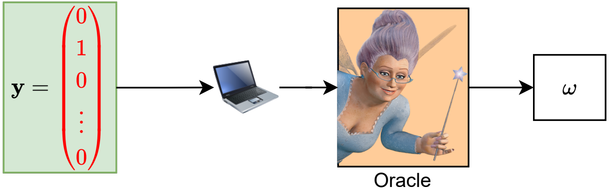
$$\mathcal{C}.\mathtt{Decode}(\mathbf{v} - \mathbf{uy})$$

Choosing → $(\mathbf{u}, \mathbf{v}) = (1, 0)$ leads to compute $\mathcal{C}.\mathtt{Decode}(\mathbf{y})$





Attack Scenario II



ω is known public parameter of HQC.

Attack Scenario III

If $\tilde{\mathbf{v}}$ has an Hamming weight of 1, they are two possibilities :

$$\tilde{\mathbf{v}} - \mathbf{y} = \begin{pmatrix}0\\1\\0\\\vdots\\0\end{pmatrix} - \begin{pmatrix}0\\1\\0\\\vdots\\0\end{pmatrix} = \begin{pmatrix}0\\0\\0\\\vdots\\0\end{pmatrix}$$

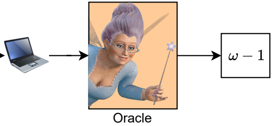


Figure – Collision Case

$$\tilde{\mathbf{v}} - \mathbf{y} = \begin{pmatrix}1\\0\\0\\\vdots\\0\end{pmatrix} - \begin{pmatrix}0\\1\\0\\\vdots\\0\end{pmatrix} = \begin{pmatrix}1\\1\\0\\\vdots\\0\end{pmatrix}$$

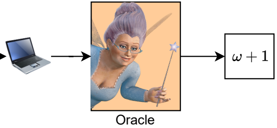
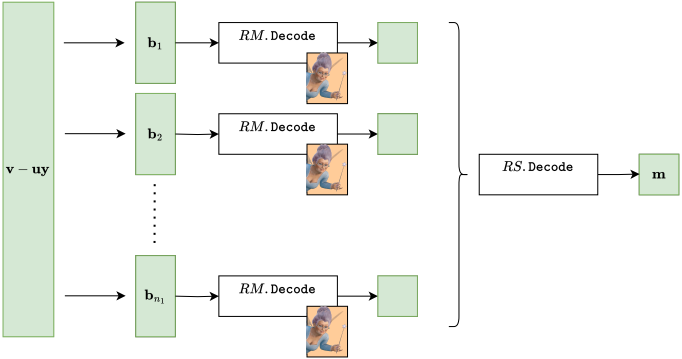


Figure – No-collision Case

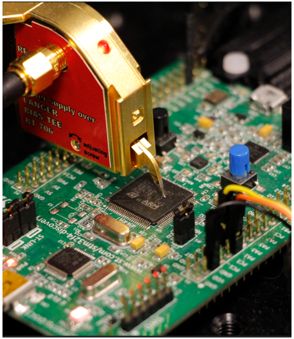
Divide and Conquer



- Each decoder manipulates a codeword of small Hamming weight (≤ 5 with probability $\geq 98\%$)

How to build the Oracle ?

$$\text{Class } i = \left\{ \mathbf{x} \xleftarrow{\$} \mathbb{F}_2^{n_2}, \mathtt{HW}(\mathbf{x}) = i \right\}$$



→ Set-Up :

- STM32F407
- Langer Near Field Probe
- Rhode-Schwarz RTO2024
- 50000 electromagnetic measurement per class.

Leakage Assessment

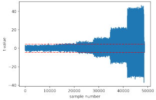
For two sets S_0 and S_1 with cardinality n_0 and n_1, means μ_0 and μ_1 and variances σ_0 and σ_1.

$$t = \frac{\mu_0 - \mu_1}{\sqrt{\left(\frac{\sigma_0^2}{n_0} + \frac{\sigma_1^2}{n_1}\right)}} \tag{1}$$

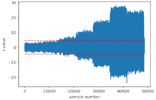
We look for absolute t-values greater than 4.5.

- If $|t| \geq 4.5$, it means that they exists a statistical difference with confidence 99.9999% that may be exploit with SCA.
- Otherwise, they are no first order distinguability to exploit.

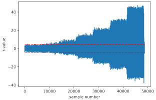
t-test Results



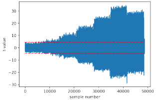
(a) Cl. 0 and 1



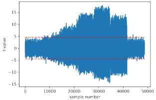
(b) Cl. 0 and 2



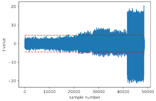
(c) Cl. 0 and 3



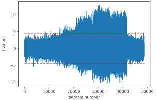
(d) Cl. 0 and 4



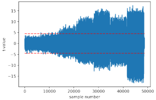
(e) Cl. 0 and 5



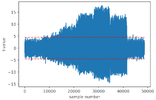
(f) Cl. 1 and 2



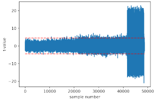
(g) Cl. 1 and 3



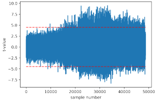
(h) Cl. 1 and 4



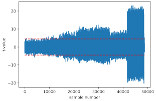
(i) Cl. 1 and 5



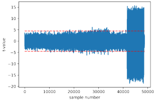
(j) Cl.2 and 3



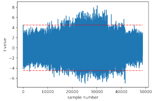
(k) Cl. 2 and 4



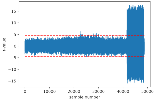
(l) Cl. 2 and 5



(m) Cl. 3 and 4



(n) Cl. 3 and 5



(o) Cl. 4 and 5

Success rate of the Oracle classification and Attack Summary

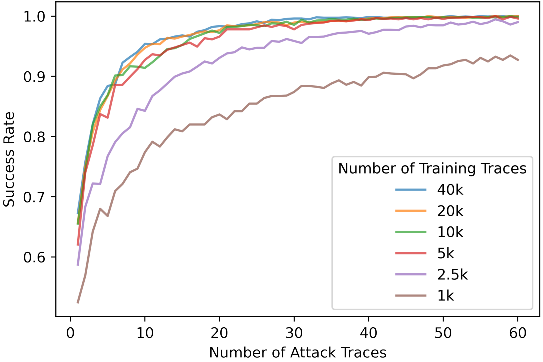


Figure – Single bit success rate recovery depending on the number of attack traces and the number of training traces per class.

Success rate of the Oracle classification and Attack Summary

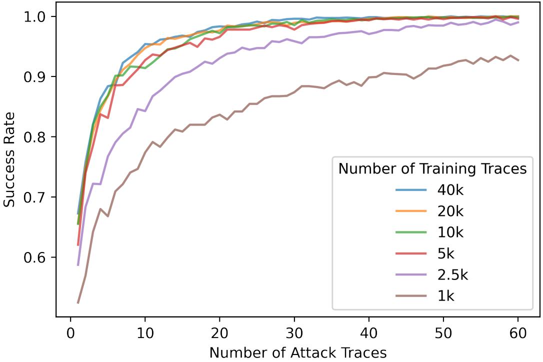


Figure – Single bit success rate recovery depending on the number of attack traces and the number of training traces per class.

Attack Summary :

- 50 attack traces are enough to obtain 100% accuracy
- Reed-Muller decoding independence
- Finally, $50 \times 384 = 19200$ traces are enough to target HQC-128.

Masking Countermeasure

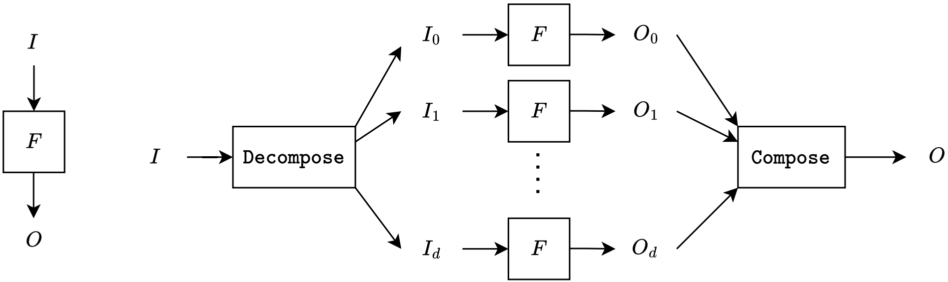


Figure – d order Masking of a linear operation F

We can apply this strategy to the Reed-Muller Decoder

- Reduce the success probability from p to p^{d+1}

Masking Countermeasure

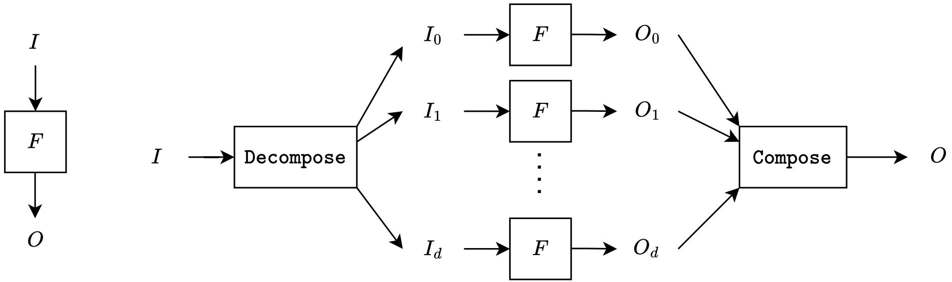
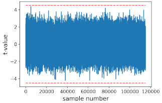


Figure – d order Masking of a linear operation F

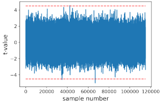
We can apply this strategy to the Reed-Muller Decoder

- Reduce the success probability from p to p^{d+1}
- Change the distribution of the inputs.

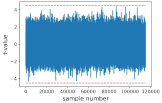
t-test Results



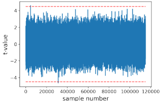
(a) Cl. 0 and 1



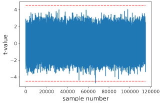
(b) Cl. 0 and 2



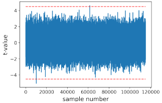
(c) Cl. 0 and 3



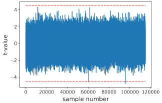
(d) Cl. 0 and 4



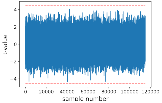
(e) Cl. 0 and 5



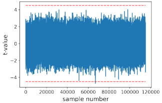
(f) Cl. 1 and 2



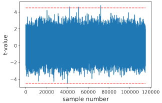
(g) Cl. 1 and 3



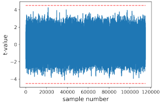
(h) Cl. 1 and 4



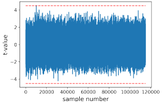
(i) Cl. 1 and 5



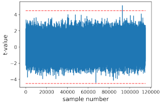
(j) Cl.2 and 3



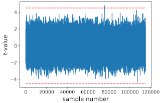
(k) Cl. 2 and 4



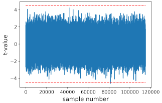
(l) Cl. 2 and 5



(m) Cl. 3 and 4



(n) Cl. 3 and 5



(o) Cl. 4 and 5

Table of Contents

Decryption Failure Rate (DFR)

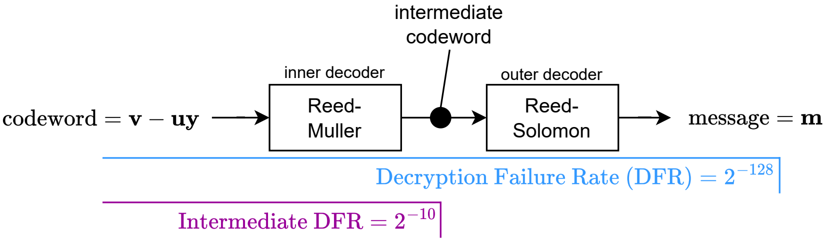
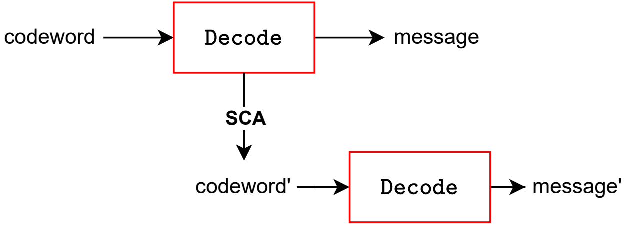


Figure – Decryption Failure Rate of HQC

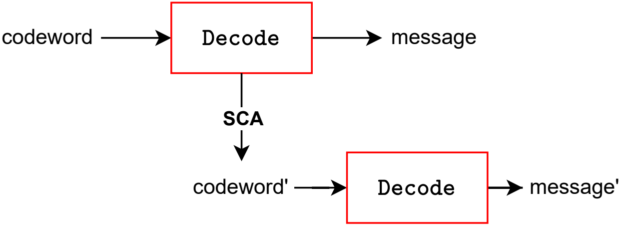
- Reed-Solomon code manipulates an error-free intermediate codeword.

Re-decoding Strategy



→ Side-channel errors correction with Error correcting codes structure !

Re-decoding Strategy



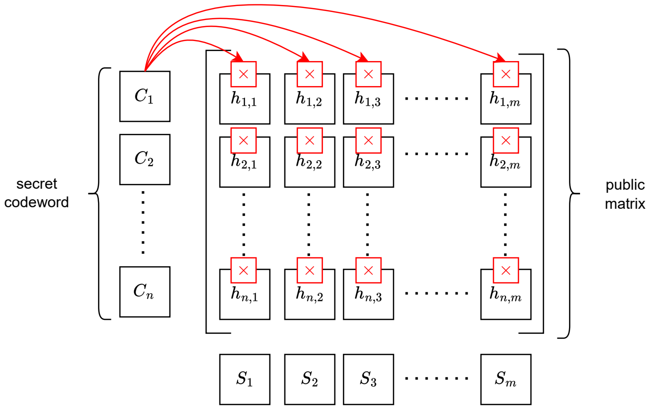
$\rightarrow$ Side-channel errors correction with Error correcting codes structure !

Security level	HQC parameters			List decoder
λ	k_1	n_1	t	τ_{GS}
HQC-128	16	46	15	19
HQC-192	24	56	16	19
HQC-256	32	90	29	36

Table – More powerful decoder for Reed-Solomon codes [VG99]

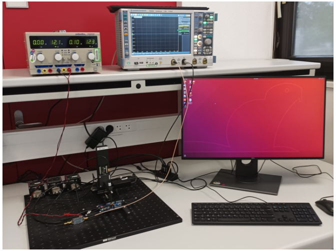
Attack Scenario – Reed-Solomon Decoder

- Target the Reed-Solomon Syndrome computation $\mathbf{Hc}^T$ to recover the codeword $\mathbf{c}$.



Attacker Model

In theory	In practice
Access to a clone device	Both training and attack on the same device
One target function only	Target the Galois field multiplication
No control on the SNR	No trace averaging (true single trace attack)



→ Set-Up :

- STM32F407
- Langer Probe
- Rhode-Schwarz RTO2024

Templates on the Galois field multiplication operands

- Galois field multiplication based on FFT strategy [BGTZ08]

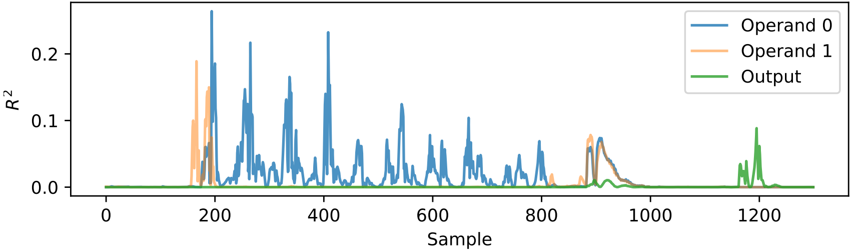


Figure – Leakage Assesment on Galois field multiplication

	Value template accuracy	Hamming weight template accuracy
Operand 0	**0.9389**	0.5929
Operand 1	0.0211	0.3035
Output	0.0221	**0.5178**

Table – Hamming weight and value templates accuracies on `gf_mul`. Each attack has been performed 400 times. 10%/90% validation/training segmentation.

- Use the 93.89% accuracy to build a straightforward attack !

	Value template accuracy	Hamming weight template accuracy
Operand 0	**0.9389**	0.5929
Operand 1	0.0211	0.3035
Output	0.0221	**0.5178**

Table – Hamming weight and value templates accuracies on `gf_mul`. Each attack has been performed 400 times. 10%/90% validation/training segmentation.

- Use the 93.89% accuracy to build a straightforward attack !
- Suppose that a wise developper swapp the two operands ($a \times b = b \times a$)
- (we keep this swapp until the end of this presentation)
- We then exploit the $51,78\%$ accuracy on the Hamming weight of the output.

	Value template accuracy	Hamming weight template accuracy
Operand 0	**0.9389**	0.5929
Operand 1	0.0211	0.3035
Output	0.0221	**0.5178**

Table – Hamming weight and value templates accuracies on `gf_mul`. Each attack has been performed 400 times. 10%/90% validation/training segmentation.

- Use the 93.89% accuracy to build a straightforward attack !
- Suppose that a wise developper swapp the two operands ($a \times b = b \times a$)
- (we keep this swapp until the end of this presentation)
- We then exploit the $51,78\%$ accuracy on the Hamming weight of the output.

How to efficiently exploit this "low accuracy" leakage ? → Belief Propagation.

Attack Description

- Message recovery attack with a single trace !
- First used of **Belief Propagation** [Mac03, KFL01] against code-based cryptography.

Idea : combine several weak physical leaks to obtain strong information

- Introduced by Veyrat-Chravrillon et al. [VCGS14] to attack AES in 2014
- Application against Kyber [PPM17, PP19, HHP^{+}21, HSST23, AEVR23]
  $\rightarrow$ Information Propagation through NTT
- Attack against hash function Keccak [KPP20] in 2020
- First BP attack against code-based cryptography [GMGL23]

Attack Description

- Message recovery attack with a single trace !
- First used of **Belief Propagation** [Mac03, KFL01] against code-based cryptography.

Idea : combine several weak physical leaks to obtain strong information

- Introduced by Veyrat-Chravrillon et al. [VCGS14] to attack AES in 2014
- Application against Kyber [PPM17, PP19, HHP+21, HSST23, AEVR23]
  $\rightarrow$ Information Propagation through NTT
- Attack against hash function Keccak [KPP20] in 2020
- First BP attack against code-based cryptography [GMGL23]

$\rightarrow$ Allows a message recovering within a few minutes

Belief Propagation – Overview

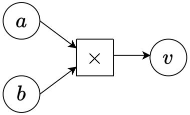


Figure – Graphical representation of a Multiplication

Belief Propagation – Overview

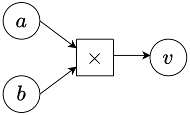


Figure – Graphical representation of a Multiplication

The Goal is to compute : $\mathbb{P}(a \mid b, v)$

Belief Propagation – Overview

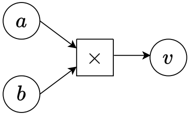


Figure – Graphical representation of a Multiplication

The Goal is to compute : $\mathbb{P}(a \mid b, v), \mathbb{P}(b \mid a, v), \mathbb{P}(v \mid a, b)$
The Marginal Probability Distributions

Belief Propagation – Overview

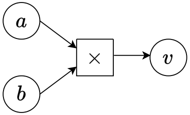


Figure – Graphical representation of a Multiplication

The Goal is to compute : $\mathbb{P}(a \mid b, v)$, $\mathbb{P}(b \mid a, v)$, $\mathbb{P}(v \mid a, b)$

The Marginal Probability Distributions

Sum Product Algorithm [KFL01] gives a solver for this problem.

$\rightarrow$ Propagate and Combine knowledge

Reed-Solomon syndrome computation graphical representation

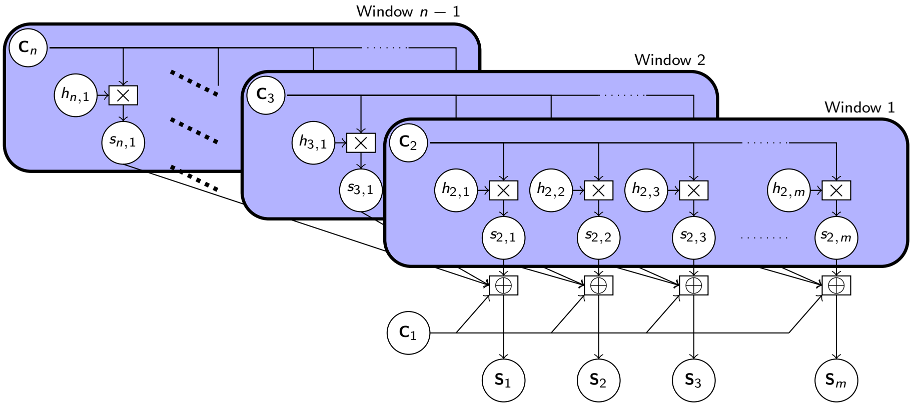


Figure – Graphical representation of the RS syndrome computation from HQC

Belief Propagation – Properties

What is proven ?

- Proof of convergence for tree like graphes
- `graph_depth` iterations are requiered to converge

Belief Propagation – Properties

What is proven ?

- Proof of convergence for tree like graphes
- `graph_depth` iterations are requiered to converge

What is not proven ?

- No proof of convergence for Cyclic graphes (oscillation phenomenon)

→ solution : Loopy Belief Propagation

Attack Accuracy in Simulation

→ Leakage on outputs of Galois field multiplication + Run BP :

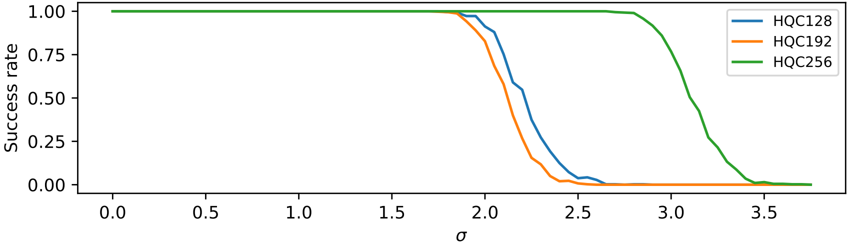


Figure – Simulated success rate of SASCA on the decoder, with re-decoding strategy, depending on the selected security level of HQC

- Attack works at high noise levels
- Attack strength increases with security level

Countermeasure ? – Codeword Masking (High Level Masking) Broken !

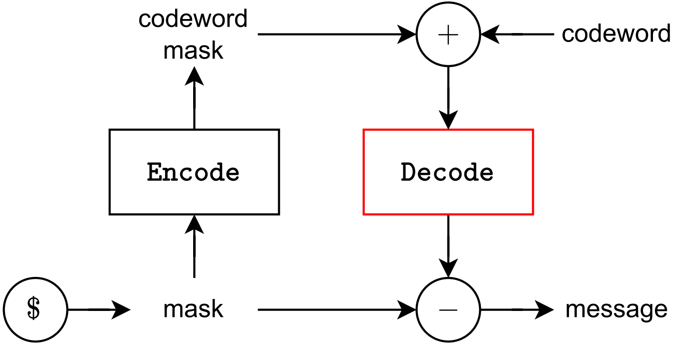


Figure – Codeword Masking [MSS13]

- Attack against the decoder which manipulates Galois field multiplications → Inefficient countermeasure

Encoder Attack Accuracy in Simulation

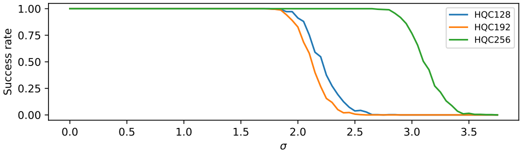


Figure – Simulated Success rate of the attack against the decoder

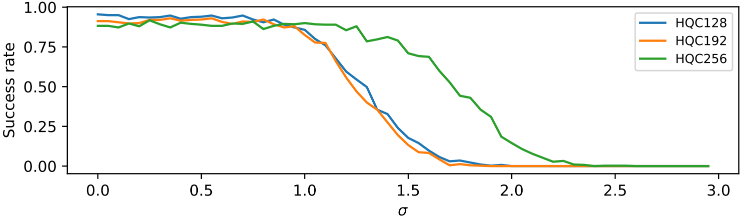


Figure – Simulated success rate of the attack against the encoder

→ Several cycles in the Encoder graph :

- Oscillation phenomenons.
- Attack less accurate at higher noise levels.

re-encryption step from HHK transform

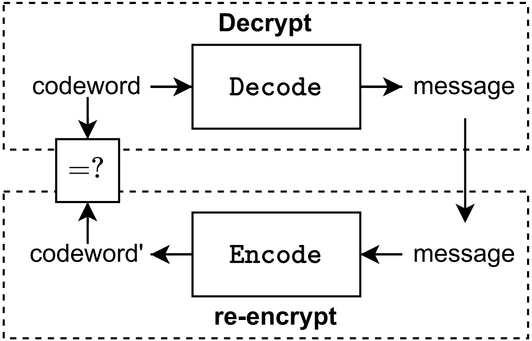


Figure – HQC Structure with HHK transform

- HQC-KEM is based on HHK transform [HHK17]
- This transform introduces a re-encryption step.

re-encryption step from HHK transform

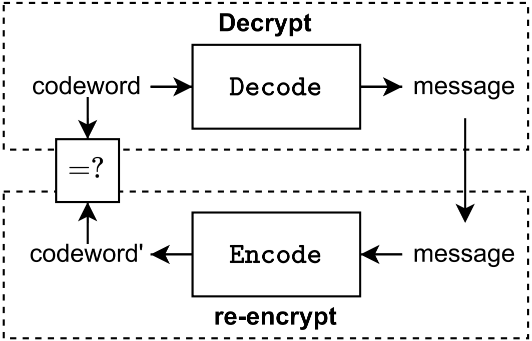


Figure – HQC Structure with HHK transform

- HQC-KEM is based on HHK transform [HHK17]
- This transform introduces a re-encryption step.

- Enable to concatenate graphs
- First attack exploiting both encryption and re-encryption

Re-encryption Attack Accuracy in Simulation

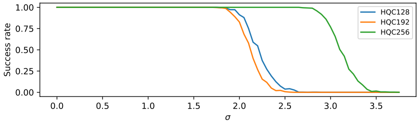


Figure – Simulated Success rate against the decoder

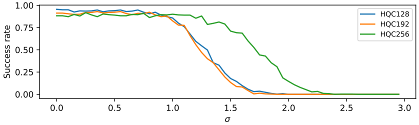


Figure – Simulated Success rate against the encoder

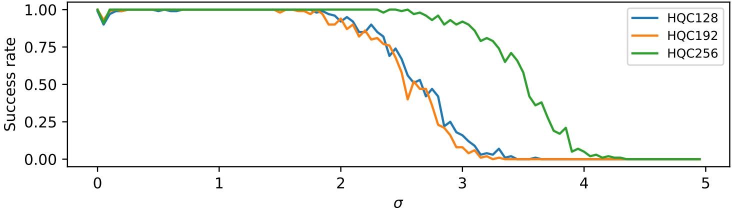


Figure – Simulated Success rate against the concatenated decoder and encoder graph

- Concatenated graph increases the strength of the attack !
- Observation of oscillation phenomenon (encoder cycles)

Re-encryption Attack Accuracy in Simulation

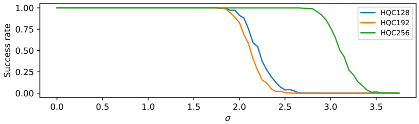


Figure – Simulated Success rate against the decoder

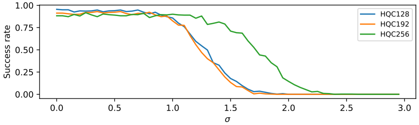


Figure – Simulated Success rate against the encoder

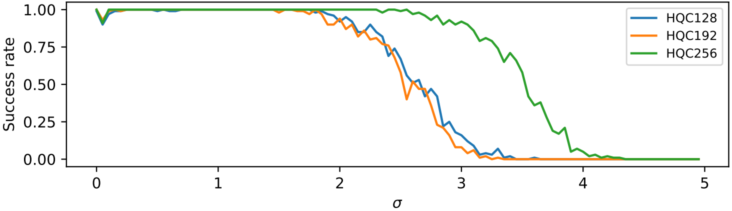


Figure – Simulated Success rate against the concatenated decoder and encoder graph

- Concatenated graph increases the strength of the attack !
- Observation of oscillation phenomenon (encoder cycles)

$\rightarrow$ Efficient shuffling countermeasure to protect the Encoder and the Decoder !

Table of Contents

- An adversary can choose a set of t wires in the circuit
- We simulate it by a perfect knowledge of the values carried by the chosen wires.
- A gadget is t-probing secure if the output of any t-probing adversary is indenpedent of sensitive data.

- An adversary can choose a set of t wires in the circuit
- We simulate it by a perfect knowledge of the values carried by the chosen wires.
- A gadget is t-probing secure if the output of any t-probing adversary is indenpedent of sensitive data.

How to build a gadget ? $\rightarrow$ We will use a low level **masking**.

- Boolean : $a = \bigoplus_{i=0}^{t} a_i$
- Artihmetic : $a = \sum_{i=0}^{t} a_i \mod q$

Gadget properties

- t-Non-Interference (t-NI)
  - $\rightarrow$ Every set of t internal probes can be simulated with at most t shares of each input.
- t-Strong Non-Interference (t-SNI)
  - $\rightarrow$ Every set I of t_1 internal probes and every set O of t_2 output probes such that $t_1 + t_2 \leq t$, the set of probes $I \cup O$ can be simulated with t_1 shares of each input.
- Probe Isolating Non-Interference (PINI)
  - $\rightarrow$ Introduces the notion of propagated probes.

Gadget properties

- t-Non-Interference (t-NI)
  - → Every set of t internal probes can be simulated with at most t shares of each input.
- t-Strong Non-Interference (t-SNI)
  - → Every set I of t_1 internal probes and every set O of t_2 output probes such that $t_1 + t_2 \leq t$, the set of probes $I \cup O$ can be simulated with t_1 shares of each input.
- Probe Isolating Non-Interference (PINI)
  - → Introduces the notion of propagated probes.

Interferences and probes propagations can be prevented by refreshing the shares.

Mask Refresh

```c
/**
 * @brief Refresh 8b additives shares with n number of shares
 *
 * @param[out] x masked value. overwritten.
  * @param[in] n Number of shares
 */
void arith_8b_refresh(uint8_t *x, int n)
{
    for(size_t i = 0; i < n; ++i)
    {
        for(size_t j = i+1; j < n; ++j)
        {
            uint8_t r = next_8();
            x[0] += r;
            x[j] -= r;
        }
    }
}
```

Figure – Refresh algorithm

- Complexity of $\mathcal{O}(d^2)$.
- Requiered to prevent Interferences !

Low level masking

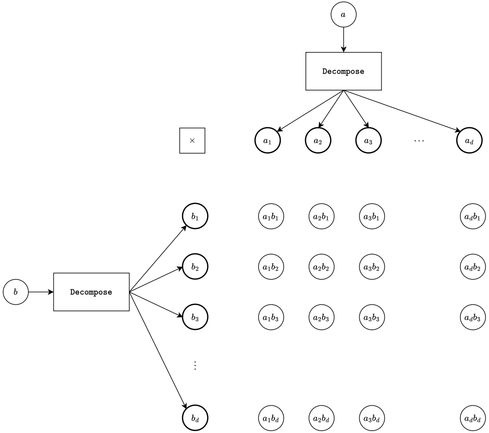


Figure – Low level Masking of a multiplication $\times$ with d shares

Low level masking 2

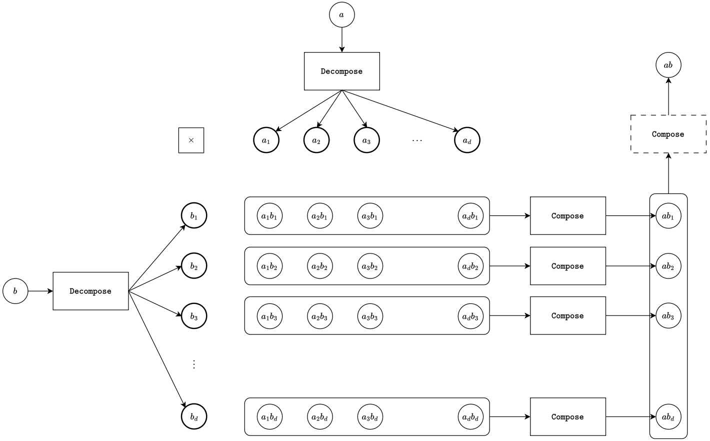


Figure – Low level Masking of a multiplication $\times$ with d shares

Low level masking 3

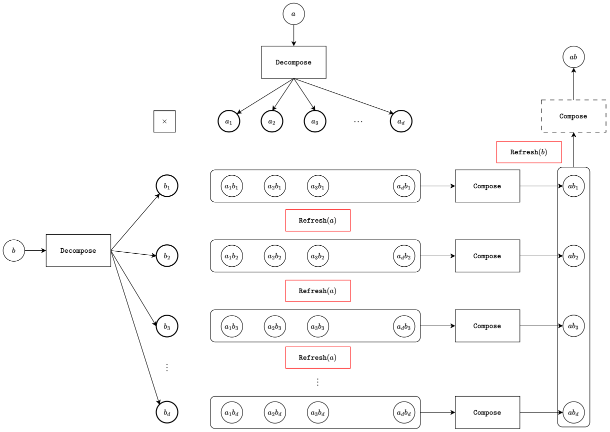


Figure – Low level Masking of a multiplication $\times$ with d shares

Masked Reed-Solomon Encoder

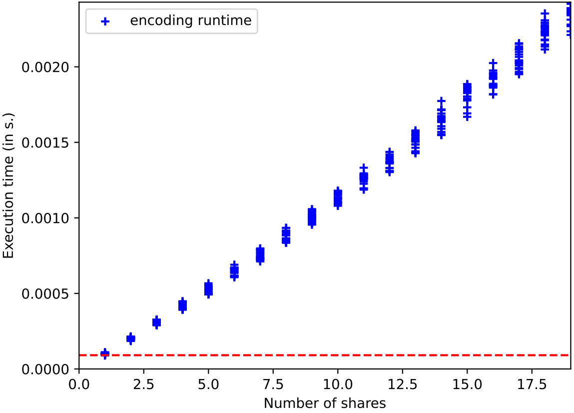


Figure – Average running time of HQC RS encoder

Masked Reed-Solomon Decoder

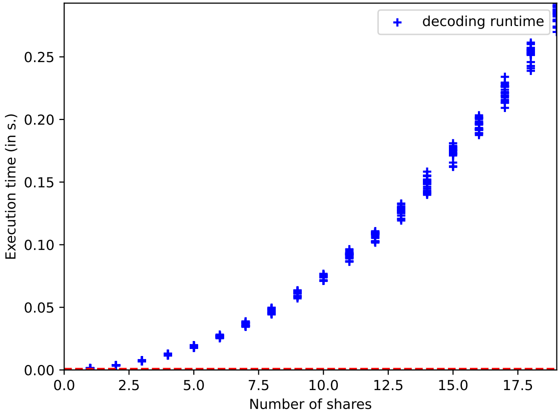


Figure – Average running time of HQC RS decoder

HQC RS running times

Number of shares	$\emptyset$	1	2	4	8	16
HQC RS Encoder	1	1.096	2.227	4.569	9.767	20.962
HQC RS Decoder	1	15.586	41.074	135.080	520.424	2148.040

Table – Reed-Solomon Encoder and decoder running times with reference implementation as refrence [AMAB+23]

- Cost of masking is at least a factor d, with d number of shares.
- But refresh cost $\mathcal{O}(d^2)$.
- The structure of gadgets can dramatically lower the performance.

Table of Contents

Conclusions and Persecpectives

- Side-Channel Attacks represents a threat for (PQ) cryptography
- Error Correcting Codes Structure can be exploit for Side-Channel purposes

Work In Progress

- Secure HQC against side-channel attacks [ABC+22, DR24]

Future Works

- Secured PQC Schemes against SCA (Fully-masking) → MPC-in-the-head schemes [ABB+24, MFG+23]

Conclusions and Persecpectives

- Side-Channel Attacks represents a threat for (PQ) cryptography
- Error Correcting Codes Structure can be exploit for Side-Channel purposes

Work In Progress

- Secure HQC against side-channel attacks [ABC$^+$22, DR24]

Future Works

- Secured PQC Schemes against SCA (Fully-masking) $\rightarrow$ MPC-in-the-head schemes [ABB$^+$24, MFG$^+$23]

Thank you for your attention !
Any questions ?

guillaume.goy@unilim.fr

References I

Nicolas Aragon, Paulo Barreto, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Shay Gueron, Tim Guneysu, Carlos Aguilar Melchor, et al.
BIKE : Bit Flipping Key Encapsulation.
2017.

Gora Adj, Stefano Barbero, Emanuele Bellini, Andre Esser, Luis Rivera-Zamarripa, Carlo Sanna, Javier Verbel, and Floyd Zweydinger.
Mirith : Efficient post-quantum signatures from minrank in the head.
IACR Transactions on Cryptographic Hardware and Embedded Systems, 2024(2) :304–328, 2024.

Melissa Azouaoui, Olivier Bronchain, Gaëtan Cassiers, Clément Hoffmann, Yulia Kuzovkova, Joost Renes, Markus Schönauer, Tobias Schneider, François-Xavier Standaert, and Christine van Vredendaal.
Protecting dilithium against leakage : Revisited sensitivity analysis and improved implementations.
Cryptology ePrint Archive, 2022.

Guilhèm Assael, Philippe Elbaz-Vincent, and Guillaume Reymond.
Improving single-trace attacks on the number-theoretic transform for cortex-m4.
In *2023 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 111–121. IEEE, 2023.

Carlos Aguilar-Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, and Gilles Zémor.
Hamming Quasi-Cyclic (HQC).
2017.

Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Maxime Bros, Couvreur Alain, Jean-Christophe Deneuville, Philippe Gaborit, Adrien Hauteville, and Gilles Zémor.
Rank quasi-cyclic (rqc).
2020.

References II

Carlos Aguilar-Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, and Gilles Zémor.
HQC reference implementation, April, 2023.
`https://pqc-hqc.org/implementation.html`.

Daniel J Bernstein, Tung Chou, Tanja Lange, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, et al.
Classic McEliece : conservative code-based cryptography.

Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé.
CRYSTALS-Kyber : a CCA-secure module-lattice-based KEM.
In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 353–367. IEEE, 2018.

Richard P Brent, Pierrick Gaudry, Emmanuel Thomé, and Paul Zimmermann.
Faster multiplication in GF(2)[x].
In *Algorithmic Number Theory : 8th International Symposium, ANTS-VIII Banff, Canada, May 17-22, 2008 Proceedings 8*, pages 153–166. Springer, 2008.

Daniel J Bernstein, Andreas Hülsing, Stefan Kölbl, Ruben Niederhagen, Joost Rijneveld, and Peter Schwabe.
The sphincs+ signature framework.
In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 2129–2146, 2019.

Chloé Baïsse, Antoine Moran, Guillaume Goy, Julien Maillard, Nicolas Aragon, Philippe Gaborit, Maxime Lecomte, and Antoine Loiseau.
Secret and shared keys recovery on hamming quasi-cyclic with sasca.
Cryptology ePrint Archive, 2024.

References III

Elwyn Berlekamp, Robert McEliece, and Henk Van Tilborg.
On the inherent intractability of certain coding problems (corresp.).
IEEE Transactions on Information Theory, 24(3) :384–386, 1978.

Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé.
Crystals-dilithium : A lattice-based digital signature scheme.
IACR Transactions on Cryptographic Hardware and Embedded Systems, pages 238–268, 2018.

Loïc Demange and Mélissa Rossi.
A provably masked implementation of bike key encapsulation mechanism.
Cryptology ePrint Archive, 2024.

Guillaume Goy, Antoine Loiseau, and Philippe Gaborit.
A new key recovery side-channel attack on HQC with chosen ciphertext.
In *International Conference on Post-Quantum Cryptography*, pages 353–371. Springer, 2022.

Guillaume Goy, Antoine Loiseau, and Phlippe Gaborit.
Estimating the strength of horizontal correlation attacks in the hamming weight leakage model : A side-channel analysis on HQC KEM.
In *WCC 2022 : The Twelfth International Workshop on Coding and Cryptography*, page WCC_2022_paper_48, 2022.

Guillaume Goy, Julien Maillard, Philippe Gaborit, and Antoine Loiseau.
Single trace HQC shared key recovery with SASCA.
Cryptology ePrint Archive, 2023.
https://ia.cr/2023/1590.

References IV

Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz.
A modular analysis of the fujisaki-okamoto transformation.
In *Theory of Cryptography Conference*, pages 341–371. Springer, 2017.

Mike Hamburg, Julius Hermelink, Robert Primas, Simona Samardjiska, Thomas Schamberger, Silvan Streit, Emanuele Strieder, and Christine van Vredendaal.
Chosen ciphertext k-trace attacks on masked CCA2 secure kyber.
IACR Transactions on Cryptographic Hardware and Embedded Systems, pages 88–113, 2021.

Julius Hermelink, Silvan Streit, Emanuele Strieder, and Katharina Thieme.
Adapting belief propagation to counter shuffling of NTTs.
IACR Transactions on Cryptographic Hardware and Embedded Systems, pages 60–88, 2023.

Frank R Kschischang, Brendan J Frey, and H-A Loeliger.
Factor graphs and the sum-product algorithm.
IEEE Transactions on information theory, 47(2) :498–519, 2001.

Neal Koblitz.
Elliptic curve cryptosystems.
Mathematics of computation, 48(177) :203–209, 1987.

Paul C Kocher.
Timing attacks on implementations of diffie-hellman, RSA, DSS, and other systems.
In *Advances in Cryptology—CRYPTO'96 : 16th Annual International Cryptology Conference Santa Barbara, California, USA August 18–22, 1996 Proceedings 16*, pages 104–113. Springer, 1996.

References V

Matthias J Kannwischer, Peter Pessl, and Robert Primas.
Single-trace attacks on keccak.
Cryptology ePrint Archive, 2020.

David JC MacKay.
Information theory, inference and learning algorithms.
Cambridge university press, 2003.

Robert J McEliece.
A public-key cryptosystem based on algebraic.
Coding Thv, 4244 :114–116, 1978.

C Aguilar Melchor, Thibauld Feneuil, Nicolas Gama, Shay Gueron, James Howe, David Joseph, Antoine Joux, Edoardo Persichetti, Tovohery H Randrianarisoa, Matthieu Rivain, et al.
Sdith.
NIST Round 1 submission to the Additional Call for Signature Schemes, 2023.

Victor S Miller.
Use of elliptic curves in cryptography.
In *Conference on the theory and application of cryptographic techniques*, pages 417–426. Springer, 1985.

Dominik Merli, Frederic Stumpf, and Georg Sigl.
Protecting PUF error correction by codeword masking.
Cryptology ePrint Archive, 2013.

References VI

Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang.
Falcon.
Post-Quantum Cryptography Project of NIST, 2020.

Peter Pessl and Robert Primas.
More practical single-trace attacks on the number theoretic transform.
In *Progress in Cryptology–LATINCRYPT 2019 : 6th International Conference on Cryptology and Information Security in Latin America, Santiago de Chile, Chile, October 2–4, 2019, Proceedings 6*, pages 130–149. Springer, 2019.

Robert Primas, Peter Pessl, and Stefan Mangard.
Single-trace side-channel attacks on masked lattice-based encryption.
In *Cryptographic Hardware and Embedded Systems–CHES 2017 : 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, pages 513–533. Springer, 2017.

Ronald L Rivest, Adi Shamir, and Leonard Adleman.
A method for obtaining digital signatures and public-key cryptosystems.
Communications of the ACM, 21(2) :120–126, 1978.

Thomas Schamberger, Lukas Holzbaur, Julian Renner, Antonia Wachter-Zeh, and Georg Sigl.
A power side-channel attack on the reed-muller reed-solomon version of the HQC cryptosystem.
In *International Conference on Post-Quantum Cryptography*, pages 327–352. Springer, 2022.

Nicolas Veyrat-Charvillon, Benoît Gérard, and François-Xavier Standaert.
Soft analytical side-channel attacks.
In *Advances in Cryptology–ASIACRYPT 2014 : 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, ROC, December 7-11, 2014. Proceedings, Part I 20*, pages 282–296. Springer, 2014.

References VII

Madhu Sudan Venkatesan Guruswami.
Improved decoding of Reed-Solomon and algebraic-geometry codes.
IEEE Transactions on Information Theory, 45(6) :1757–1767, 1999.

Detecting Collisions

If $\mathbf{v}$ has an Hamming weight of 1, they are two possibilities :

1. $\mathrm{Supp}(\mathbf{y}) \cap \mathrm{Supp}(\mathbf{v}) = \mathrm{Supp}(\mathbf{v})$. Then $\mathtt{HW}(\mathbf{v} - \mathbf{y}) = \mathtt{HW}(\mathbf{y}) - 1$, the decoder will correct one error less than the reference decoding of $\mathbf{y}$.

$$\mathcal{O}_b^{\mathrm{RM}}(\mathbf{v} - \mathbf{y}) = O_b^{\mathrm{RM}}(\mathbf{y}) - 1$$

2. $\mathrm{Supp}(\mathbf{y}) \cap \mathrm{Supp}(\mathbf{v}) = \varnothing$. Then $\mathtt{HW}(\mathbf{v} - \mathbf{y}) = \mathtt{HW}(\mathbf{y}) + 1$, the decoder will correct one error more than the reference decoding of $\mathbf{y}$.

$$\mathcal{O}_b^{\mathrm{RM}}(\mathbf{v} - \mathbf{y}) = O_b^{\mathrm{RM}}(\mathbf{y}) + 1$$

- **Strategy** Remember locations where Oracle outputs 1 less than the reference value.

Divide and Conquer

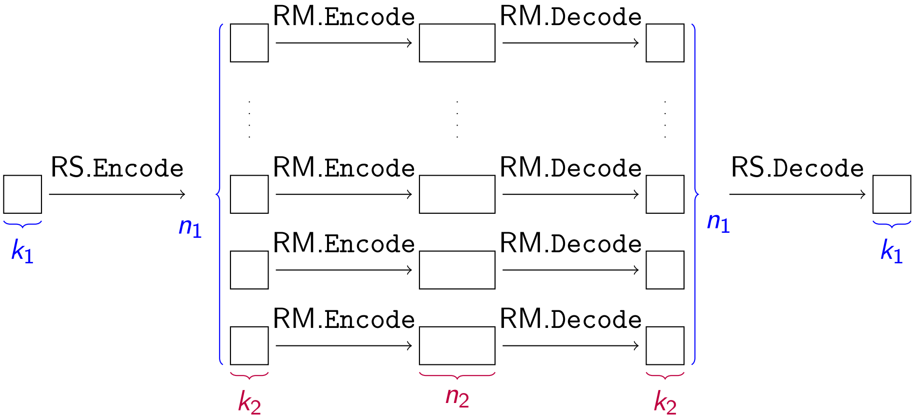


Figure – Simplified HQC Concatenated RMRS Codes Framework

Breaking shuffling countermeasures

- Fine Shuffling (Adapted from a Kyber countermeasure)
  $\rightarrow$ Randomly choose $a \times b$ or $b \times a$.
- Coarse shuffling (Adapted from a Kyber countermeasure)
  $\rightarrow$ Randomly shuffle columns of the parity check matrix

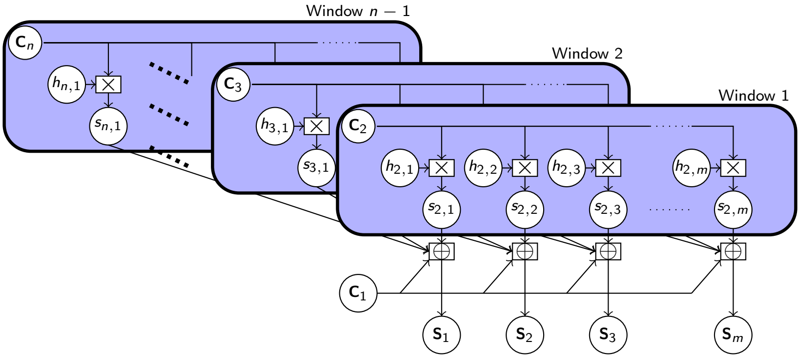
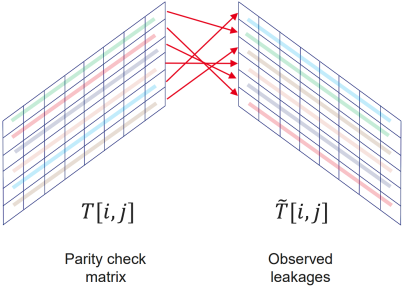


Figure – Graphical representation of the RS syndrome computation from HQC

Breaking shuffling countermeasures 2

- Window Shuffling (Novelty)

  $\rightarrow$ Randomly shuffle lines of the parity check matrix



$$D[i, i'] = \sum_{j=1}^{256} \mathtt{d}\left(\tilde{T}[i,j], T[i',j]\right)$$

Instance of the assignment Problem.

$\rightarrow$ Solver : Hungarian algorithm.

Full Shuffling Countermeasure

- Lines Shuffling $\rightarrow$ Not enough !
- Columns Shuffling $\rightarrow$ Not enough !

$\hookrightarrow$ Entire Matrix Shuffling !

$$2^{504}, \quad 2^{614}, \quad \text{and } 2^{1030}$$

- We can change the encoder to apply the same countermeasure

Reed-Solomon syndrome computation graphical representation

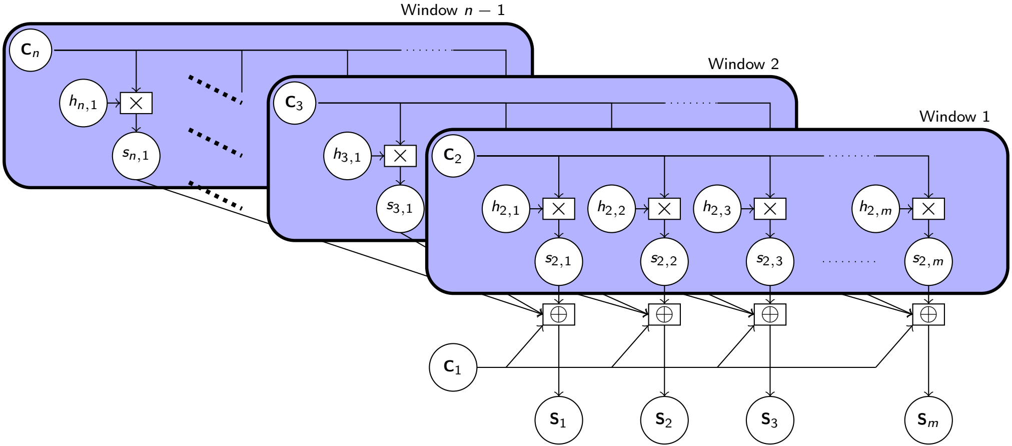


Figure – Graphical representation of the RS syndrome computation from HQC

Reed-Solomon Encoder graphical representation

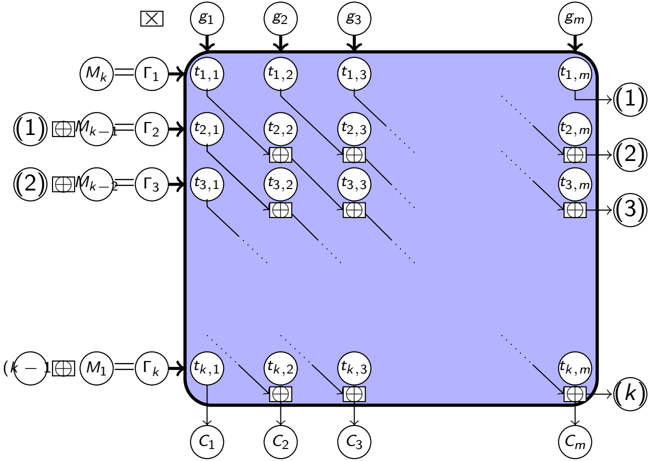


Figure – Graphical representation of the RS encoder from HQC