HQC: Hamming Quasi-Cyclic

An IND-CCA2 Code-based Public Key Encryption Scheme

November 30, 2022

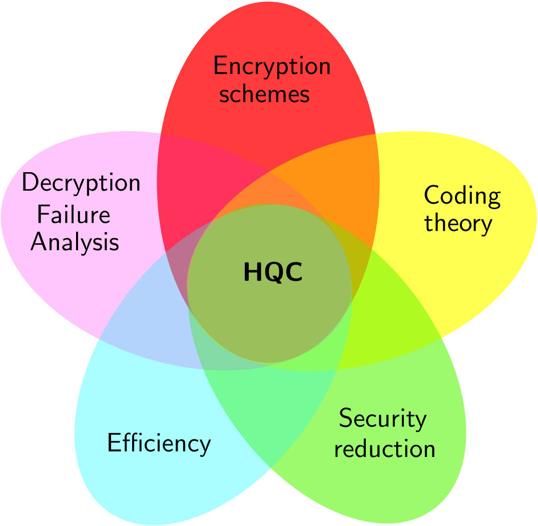
NIST 4th PQC Standardization Conference

https://pqc-hqc.org

C. Aguilar Melchor	Sandbox	A. Dion	ISAE-Supaéro, Univ. Toulouse
N. Aragon	Naquidis, France	**P. Gaborit**	University of Limoges
S. Bettaieb	Worldline	J. Lacan	ISAE-Supaéro, Univ. Toulouse
L. Bidoux	TII, UAE	E. Persichetti	Florida Atlantic University
O. Blazy	Ecole Polytechnique, France	J.-M. Robert	University of Toulon
J. Bos	Worldline	P. Véron	University of Toulon
J.-C. Deneuville	ENAC, University of Toulouse	G. Zémor	IMB, University of Bordeaux

Outline

HQC Classification / Design Rationale



Important features:

- IND-CCA2 code-based PKE
- Reduction to a well-known and difficult problem:

  Decoding random quasi-cyclic codes
- No hidden trap in the code
- Efficient decoding
- Precise DFR analysis

HQC Encryption Scheme

Encryption scheme in **H**amming metric, using **Q**uasi-**C**yclic Codes

- Notation: Secret data - Public data - One-time Randomness
- $\mathbf{G}$ is the generator matrix of some public code $\mathcal{C}$
- $\mathcal{S}_w^n(\mathbb{F}_2) = \{\mathbf{x} \in \mathbb{F}_2^n \text{ such that } \omega(\mathbf{x}) = w\}$

Alice		Bob
$\text{seed}_{\mathbf{h}} \xleftarrow{\$} \{0,1\}^\lambda$, $\mathbf{h} \xleftarrow{\text{seed}_{\mathbf{h}}} \mathbb{F}_2^n$ $\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{hy}$	$\xrightarrow{\text{seed}_{\mathbf{h}}, \mathbf{s}}$	
	$\xleftarrow{\mathbf{u}, \mathbf{v}}$	$\mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$, $\mathbf{e} \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$ $\mathbf{u} \leftarrow \mathbf{r}_1 + \mathbf{hr}_2$, $\mathbf{v} \leftarrow \mathbf{mG} + \mathbf{sr}_2 + \mathbf{e}$
$\mathbf{m} \leftarrow \mathcal{C}.\text{Decode}(\mathbf{v} - \mathbf{uy})$		

Fourth round tweaks

◇ **Multi-ciphertext attack**: Addition of a public salt value into the ciphertext, in order to counter a simple multi-ciphertext attack discussed in NIST's PQC mailing list in July 2021. Although the attack is not in the security model considered, we chose to make a modification. The randomness θ is now computed from a salt together with the public key.

$$\theta = shake256\_512(m||public\ key||salt)$$

◇ **Countermeasure to a timing attack**: In the recent paper [GHJLNS22]: the authors propose an attack on BIKE and HQC on the sampling of small weight vectors.
We modified the generation of the small vectors according a paper by N. Sendrier [Sen21] which avoids the attack at a cost of a small biais which does not impact the security.
In practice this countermeasure implies a loss of a few percentages points in our performance.

◇ **Constant time reference implementation**: we included a constant time (not optimized) reference version.

Parameters and performances

Sizes in kilo bytes, performances in kilo cycles:

	Public key size	Ciphertext size	Keygen	Encaps	Decaps	DFR
hqc-128	2,249	4,497	87	204	362	$< 2^{-128}$
hqc-192	4,522	9,042	204	465	755	$< 2^{-192}$
hqc-256	7,245	14,485	409	904	1505	$< 2^{-256}$

Optimized implementations

We thank the communauty for their support in trying to improve the security of HQC for side channel attacks and for improvements on hardware implementations of HQC.

⋄ [DXNNS22] VHDL hardware implementation which largely outperforms our HLS implementation and obtains very good performance using a small surface.

⋄ [Loi22] Efficient implementation of HQC for embedded systems (ARMv7)

⋄ **Versatility of the scheme**: possibility to consider customized versions of HQC for hardware with very light decoder (replace RS+RM with only Repetition code or Repetition code + Hamming codes for instance).

References

[DXNNS22] Sanjay Deshpande, Chuanqi Xu, Mamuri Nawan Kashif Nawaz and Jakub Szefer, Fast and Efficient Hardware Implementation of HQC, https://eprint.iacr.org/2022/1183.pdf

[GHJLNS22] Qian Guo, Clemens Hlauschek, Thomas Johansson, Norman Lahr, Alexander Nilsson, and Robin Leander Schroder, " Don't Reject This: Key-Recovery Timing Attacks Due to Rejection-Sampling in HQC and BIKE" ,CHES 2022.

[Loi22] Antoine Loiseau, ARMv7 implementation of HQC available at https://github.com/AL250125/LACI-HQC128)

[Sen21] Nicolas Sendrier, "Secure Sampling of Constant-Weight Words – Application to BIKE", https://eprint.iacr.org/2021/1631.

Questions ?

HQC official website and updates:
https://pqc-hqc.org/