HQC: **H**amming **Q**uasi-**C**yclic

An IND-CCA2 Code-based Public Key Encryption Scheme

April 10, 2024

NIST 5th PQC Standardization Conference

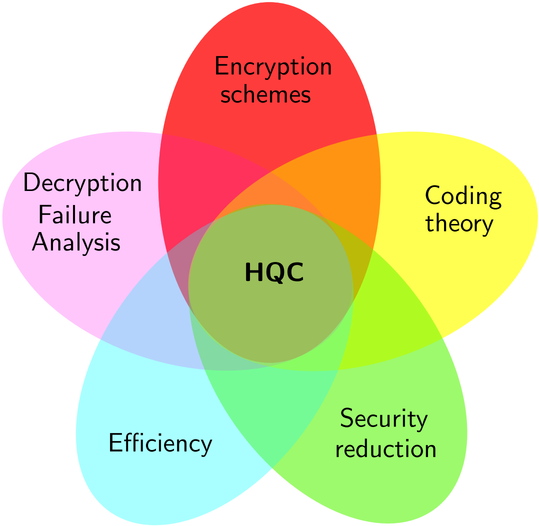
https://pqc-hqc.org

C. Aguilar Melchor	Sandbox	A. Dion	ISAE-Supaéro, Univ. Toulouse
N. Aragon	University of Limoges, France	**P. Gaborit**	University of Limoges
S. Bettaieb	TII, UAE	J. Lacan	ISAE-Supaéro, Univ. Toulouse
L. Bidoux	TII, UAE	E. Persichetti	University of Roma
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

1. HQC design rationale and recap
2. Recent updates
3. Other Optimized Implementations
4. Side-channel attacks

HQC Classification / Design Rationale



Important features:

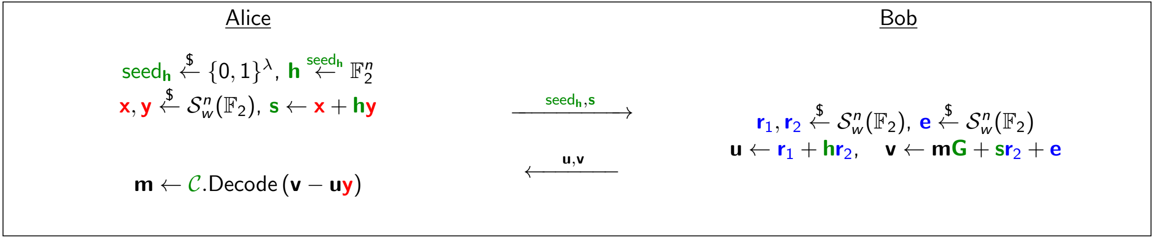
- IND-CCA2 code-based PKE
- Reduction to a well-known and difficult problem:

  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 $\mathcal{C}$
- $\mathcal{S}_w^n(\mathbb{F}_2) = \{\mathbf{x} \in \mathbb{F}_2^n$ such that $\omega(\mathbf{x}) = w\}$



Alice

$\mathsf{seed}_{\mathbf{h}} \overset{\$}{\leftarrow} \{0,1\}^{\lambda}$, $\mathbf{h} \overset{\mathsf{seed}_{\mathbf{h}}}{\leftarrow} \mathbb{F}_2^n$

$\mathbf{x}, \mathbf{y} \overset{\$}{\leftarrow} \mathcal{S}_w^n(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{hy}$

$\xrightarrow{\mathsf{seed}_{\mathbf{h}},\mathbf{s}}$

Bob

$\mathbf{r}_1, \mathbf{r}_2 \overset{\$}{\leftarrow} \mathcal{S}_w^n(\mathbb{F}_2)$, $\mathbf{e} \overset{\$}{\leftarrow} \mathcal{S}_w^n(\mathbb{F}_2)$

$\mathbf{u} \leftarrow \mathbf{r}_1 + \mathbf{hr}_2$, $\quad \mathbf{v} \leftarrow \mathbf{mG} + \mathbf{sr}_2 + \mathbf{e}$

$\xleftarrow{\mathbf{u},\mathbf{v}}$

$\mathbf{m} \leftarrow \mathcal{C}.\text{Decode}\left(\mathbf{v} - \mathbf{uy}\right)$

Recent updates

◇ **Small technical update of the security proof**: Following a comment of T. Hemmert, we did a small technical update in the IND-CPA security proof of HQC and definitions of hard problems. There was indeed an indistinguishability issue between Game 3 and 4 of the proof, coming from a lack of technical update in the proof after the introduction of truncation in Round 2. Neither the design, implementation nor parameters of the scheme are affected by this modification, only the proof is updated.

◇ **Countermeasure to a timing attack**: We fixed in the implementation a division/modulo timing-side-channel attack related to a recent paper:

Divide an Surrender: Exploiting Variable Division Instruction Timing in HQC Key Recovery attacks, by R.A. Schroder, S. Gast and Q. Guo (https://eprint.iacr.org/2024/299)

Parameters and performances (no change)

Sizes in kilobytes, performances in kilocycles:

	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	$< 2^{-256}$

Other optimized implementations

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

◇ **Green Computing**: [SFW23] Code-based Cryptography in IoT: A HW/SW Co-design of HQC: this paper shows that HQC is well adapted to green computing (best paper award in the conference IEEE 8th World Forum on Internet of Things).

◇ **Hardware implementations** several papers improved on the HardWare implementations of HQC among them:

- LEAP: Lightweight and Efficient Accelerator for Sparse Polynomial Multiplication of HQC (2023)(https://ieeexplore.ieee.org/abstract/document/10068178)
- Efficient hardware implementation of constant time sampling for HQC (2023) (https://arxiv.org/pdf/2309.16493.pdf)
- High Efficiency Hardware Design for the Post-Quantum KEM Hamming Quasi-Cyclic (HQC) cryptosystem (HOST2024)

Conclusion of the second paper: "Based on these changes, the performance gap between code-based (HQC) and lattice-based algorithms substantially decreases - from a factor of 10.5 to a factor of 1.7 in terms of execution time. To conclude, HQC is a very competitive alternative to KYBER in case it turns out to be insecure in the future in terms of resource efficiency."

◇ **TLS comparison** Several papers analyze and compare the different options for Post-Quantum TLS, among them:

- SoK: Post-Quantum TLS Handshake (https://eprint.iacr.org/2023/1873.pdf)
- Performance Impact of PQC KEMs on TLS 1.3 under varying networks characteristics (ISC 2023)

Conclusion: the efficiency depends on the data rate, under normal conditions larger key size do not seem to be an issue. Moreover measurements (of first paper) show that the TLS 1.3 handshake with HQC at security level 1 takes 1.45 the time of the classical handshake with x25519 (ECDHE) which is close to KYBER, confirming results of other previous papers.

Side-channel attacks

◇ **Side-channel attacks:** Recent papers propose new side-channel attacks on HQC, among them:

- Single trace HQC shared key recovery with SASCA, TCHES 2024
- Secret and shared key recovery on HQC with SASCA (https://eprint.iacr.org/2024/440)

All side-channel attacks require specific countermeasures, which may have a cost. Recent developments show that it is possible to increase side-channel resistance with high order masking with the use of Gadgets.

These approaches may have a significant overhead in terms of implementation, depending on the order of protection, we are working on it for HQC.

This cost shows the importance of having efficient performances for a cryptosystem since probably, secure implementations will use this type of masking in the future, which comes with a significant overhead.

Take away on HQC

- the system is mature and stable
- the decapsulation is very fast
- the DFR analysis is precise and well studied
- the inherent difficult problems are very well known
- the impact of a relatively larger key seems to be essentially non-significant in some applications such as TLS

Questions ?

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