

“Preview Writeup”: In anticipation of a package submission to the NIST Threshold Call

Title: HQC-SSD (Hamming Quasi-Cyclic Shamir Shared Decoding)

Version: Preview Writeup 1.0 (2026-10-12)¹

Team name: HQC-SSD

Team members: Jean-Christophe Deneuville, Nathan Maillet, Vincent Migliore, Cyrius Nugier

Abstract: This document previews HQC’s Decryption thresholdized using Shamir’s Sharing, with HQC’s parameters optimized in consequence (HQC-SSD (Hamming Quasi-Cyclic Shamir Shared Decoding)), a thresholdization of the decryption function performed by the NIST “to be standardized” cryptosystem HQC (Hamming Quasi-Cyclic). The security of HQC is based on the hardness of the Syndrome Decoding problem for quasi-cyclic codes. The bits of the secret key are hidden using Shamir’s Secret Sharing Scheme and given to N parties, such that any subset of at least T parties may collaborate to decrypt a message in a constant number of round in N , while an attacker can compromise at most $N-T$ parties in an adaptive way. This sharing allows a secure implementation in the byzantine model with honest majority and security in the gossip model with dishonest majority.

This submission aims to assemble and optimize some well-known and some more recent gadgets in order to first achieve an IND-CPA (Indistinguishability against Chosen Plaintext Attacks) scheme. Options are explored to achieve IND-CCA (Indistinguishability against Chosen Ciphertext Attacks) security. To this end, four parameter sets will be provided to fit different use cases and NIST-defined categories. A N2 parameter set achieves interoperability with HQC, matching KATs (Known Answer Tests) used during the standardization process. The three other parameter sets take advantage of the freedom given in the S2 category to propose use-case driven improvements. To the best of our knowledge, this is the only submission in the NIST Multi-Party Threshold Cryptography call that is code-based and meeting the N2 and S2 categories.

Proposed crypto-systems:

- (I) HQC-SSD: Hamming Quasi-Cyclic Shamir Shared Decryption (Categories N2, S2)

Keywords: Threshold Cryptography; NIST Threshold Call; Code-based Cryptography; HQC



Document License: CC BY 4.0 International

¹Version submitted to NIST-MPTC for publication

Preview writeup. This document is provided to NIST for online publication, to foster public awareness and support public discussion within the scope of the NIST First Call for Multi-Party Threshold Schemes [NIST-IR8214C]. This “preview writeup” represents a good-faith plan for a subsequent “package submission”. However, until the deadline for package submission, the team may still modify its own composition and the submission plan, including possible changes to the technical scope, and/or the used techniques or achieved results.

Team members: Jean-Christophe Deneuville^{i1,a1}, Nathan Maillet^{i2,a2,a3}, Vincent Migliore^{i3,a2,a4}, Cyrius Nugier^{i4,a1}

Open Researcher and Contributor Identifiers (ORCID):

i1 (0000-0002-5128-6729); i2 (0009-0007-8957-3567); i3 (0000-0002-6834-4235); i4 (0000-0003-1276-0296)

Affiliations:

^{a1} École Nationale de l'Aviation Civile, Toulouse, France

^{a2} Laboratoire d'Analyse et d'Architecture des Systèmes – Centre National de la Recherche Scientifique, Toulouse, France

^{a3} Électricité De France, Research & Development, Palaiseau, France

^{a4} Institut National des Sciences Appliquées, Toulouse, France

Main contacts:

- **Team mailing list:** team-thqc[at]lists[dot]recherche[dot]enac[dot]fr
- **Primary technical contact person:** Jean-Christophe Deneuville, jean-christophe[dot]deneuville[at]enac[dot]fr
- **Secondary contact person:** Cyrius Nugier, cyrius[dot]nugier[at]enac[dot]fr

Produced by humans. The team hereby confirms that the content in this preview writeup: (i) was produced by the team members, and (ii) was not produced by generative artificial intelligence (AI), with the possible exception of AI-proposed grammar improvements, minor integrated suggestions, or some well-identified and short localized portions of auxiliary content (e.g., some illustration); and (iii) was proofread by the team members.

1. Introduction

Threshold public key encryption is a type of cryptographic primitive that is similar to public key encryption (PKE), except that the private key is securely divided into shares that are distributed among N participants. In such a threshold scheme, at least a certain number T of the participants is required to successfully decrypt a ciphertext. Any set of at most $T - 1$ parties should not be able to learn any information about the plaintext.

To set up a threshold scheme, key distribution can be done with a theoretical trusted third party ‘sharing’ the decryption key and sending the shares to the parties. The parties receive the ciphertext and can then communicate computation results between each other through authenticated and confidential channels, and also through a reliable broadcast channel, ensuring that all honest parties received the same data. Finally, shares of the results are sent back to the trusted third party for recomposition into the plaintext.

In this document, we introduce HQC-SSD (Hamming Quasi-Cyclic Shamir Shared Decoding), a variant of HQC adapted to the threshold setting with constant round complexity. The core of this thresholdization is the design of a dedicated MPC protocol for the decryption algorithm of the HQC.PKE component of the HQC cryptosystem [GAABB+26]. Such a protocol has been introduced in [GLOV26a]. The resulting interactive threshold cryptosystem achieves CPA security. To achieve CCA security, we plan on using either a Fujisaki-Okamoto [FO99] or a Naor-Yung [NY90] based conversion. Depending on the computational cost, the former may require a sharing conversion or an adjustment in the HQC variables generation by changing the XOF function used. The latter requires the design of a NIZK proof system adapted for HQC ciphertext, as well as a modification of the original HQC parameters and/or design in order to make this conversion applicable.

2. Specification

2.1. Organization

This preview proposes a thresholdized HQC cryptosystem that reaches CPA security, and explores options to reach CCA security. Multiple parameter sets will be studied in order to reduce computational and communication bottlenecks and optimized for different use-cases.

2.1.1. The HQC cryptosystem

Hamming Quasi-Cyclic (HQC) is a code-based public key encryption (PKE) scheme that uses public codes: random quasi-cyclic (QC) codes for key and message security, and a code $\mathcal{C}$ for encoding and decoding, consisting of a Reed-Solomon concatenated with a duplicated Reed-Muller code [GAABB+26]. The IND-CCA2 key encapsulation mechanism (KEM) version obtained through the SFO transform with implicit rejection ($\text{SFO}^\perp$) was selected for standardization by NIST in March 2025 (to be FIPS 207).

In a nutshell, the key generation algorithm generates a public random QC code defined by $\mathbf{h} \xleftarrow{\$} \mathcal{R} := \mathbb{F}_2[X]/\langle X^n - 1 \rangle$, two private moderately sparse vectors $\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{S}_w^n := \{\mathbf{v} \in \mathcal{R} \text{ such that } |\mathbf{v}| = w\}$, and the public syndrome of this private vector $\mathbf{s} = \mathbf{x} + \mathbf{h}\mathbf{y}$. The keys are $(\text{pk}, \text{sk}) \leftarrow ((\mathbf{h}, \mathbf{s}), (\mathbf{x}, \mathbf{y}))$.

The encryption algorithm of HQC.PKE works similarly as in McEliece: the message is first encoded, then an error is added on top of this encoding. However there are two major differences between HQC and the McEliece framework: first, the code used to encode the message (as well as its associated decoding algorithm) is public; second, the added noise is way above the correction capability of the code. Only the knowledge of the private key allows removing sufficient noise to get back under this correction capability and, in turn, correctly decrypt with overwhelming probability.

Formally, the message $\mathbf{m}$ is encoded via the public code $\mathcal{C}$. We denote this concatenated encoding $\mathbf{m}\mathbf{G} \in \mathbb{F}_2^{n_1 \times n_2}$, where $\mathbf{G}$ would represent the generator matrix of the concatenated RS code (of length n_1) with duplicated RM code (of length n_2). Encryption works by generating $\mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}_{w_r}^n$ and $\mathbf{e} \xleftarrow{\$} \mathcal{S}_{w_e}^n$, computing $\mathbf{u} \leftarrow \mathbf{r}_1 + \mathbf{h}\mathbf{r}_2$ along with the structured loud noise on top of the encoded message: $\mathbf{v} \leftarrow \mathbf{m}\mathbf{G} + \mathbf{s}\mathbf{r}_2 + \mathbf{e}$. The ciphertext is $(\mathbf{u}, \mathbf{v})$.

The PKE version of the decryption also works in two steps: first, it uses the secret key to remove a lot of noise, then it applies the decoding algorithm. The noisy codeword obtained in the first step is denoted $\mathbf{t} = \mathbf{v} - \overline{\mathbf{y}}\mathbf{u}^{(\ell)} = \mathbf{c} + \mathbf{e}'$, where the $\overline{\cdot}^{(\ell)}$ operator denotes the truncation of the $\ell = n - n_1n_2$ last bits. This yields a codeword $\mathbf{c} \in \mathcal{C}$ expected to match $\mathbf{m}\mathbf{G}$ assuming $\mathbf{e}' = \mathbf{x}\mathbf{r}_2 - \mathbf{y}\mathbf{r}_1 + \mathbf{e}$ can be corrected by $\mathcal{C}$. The second step performs this decoding to retrieve $\mathbf{m}$.

The KEM decapsulation, in particular with the $\text{SFO}^\perp$ transform implies a re-encryption step, that verifies that the ciphertext was correctly formed by re-sampling the randomness and recomputing $\mathbf{u}$ and $\mathbf{v}$.

2.1.2. Thresholdizing decryption

By distributing the secret key among parties using a secret sharing technique, they can achieve threshold decryption by running a protocol instantiating the decryption functionality. The shares of the decryption key constitute the private input of the decryption protocol and the ciphertext is a public input (i.e. known by all the parties).

The core idea for the thresholdization of HQC.PKE is as follows: the computation of the shares of a noisy codeword of the message is linear in the shares of the secret key ($\mathbf{v} - \overline{\mathbf{y}}\mathbf{u}^{(\ell)} = [\mathbf{c} + \mathbf{e}']$) and thus the thresholdized version of this operation does not require communication between the parties. However, decoding a shared noisy codeword for a public code $\mathcal{C}$ is intricate. This procedure depends on the code $\mathcal{C}$ itself, the decoding procedure used, and must follow strong security guarantees in order to ensure that the protocol does not give out any information about the shares of the decryption key. A dedicated constant-round protocol is designed for distributed decryption of HQC.PKE ciphertexts [GLOV26a], instantiating $\mathcal{C}$ as in the standard HQC specifications, i.e. with a concatenated code built from a Reed-Muller code and a Reed-Solomon code.

Following [BGW88] it is possible after the communication rounds to verify the subsequent computation by identifying malicious parties and replacing the value of their shares with publicly known values (typically 0). Using this construction the scheme provides robustness and can reach full decryption despite unavailable or malicious parties.

As this technique does not fundamentally modify HQC.PKE to achieve thresholdization, this threshold variant relies on the same type of hardness assumptions as the original cryptosystem, aside from the additional assumptions for enabling MPC.

2.1.3. Reaching CCA security

Designing a multi-party protocol for decryption of HQC.PKE as described above only achieves CPA security. In order to achieve CCA security, two possible adaptations may be explored:

- One could design an MPC protocol for the entire decapsulation of the HQC.KEM, which is the Fujisaki-Okamoto transform of HQC.PKE. However, this requires being able to regenerate the sparse vectors $\mathbf{r}_1, \mathbf{r}_2, \mathbf{e}$ from the message. In the random oracle model, this can be done either: 1) by hashing the message into a seed with threshold operations, and recomposing the shares of the seed before expanding it into the vectors to recompute the ciphertext or; 2) using a threshold XOF function to replace HQC's initial randomness. Both options would require multiple rounds of communication and an important increase in the overall computation complexity.
- Instead, it is also possible to use some variant of the Naor-Yung transform [NY90; Lin03]. This solution would require each participant to verify a NIZK proof sent along the ciphertext, in order to verify that an adversary would not submit a ciphertext that would lead to a decryption failure.

A solution based on this principle was presented in order to build a threshold CCA variant of the Classic McEliece scheme [GLOV26b]. As the CPA component of the Classic McEliece is perfectly correct and deterministic, ensuring that a ciphertext is “well-formed” suffices to achieve CCA security.

However, HQC.PKE is not deterministic and has a decoding failure rate (so-called DFR, required to be negligible for HQC.KEM) that depends on the randomness of the secret key, as well as the randomness sampled during encryption. In this context, the Naor-Yung transform is harder to apply. Informally, an attacker could produce “well-formed” ciphertexts (i.e. using $\mathbf{r}_1, \mathbf{r}_2, \mathbf{e}$ of the correct Hamming weight) using a biased encryption randomness, producing more decoding errors. This bias and the observation of success or failure could help an adversary learn information about the private key.

To mitigate this issue, our proposal includes new parameter sets that set the DFR of HQC.PKE to 0. This effectively makes HQC.PKE a perfectly correct scheme, and allows applying the transform in [GLOV26b]. This is the first requirement to use the Naor-Yung in order to obtain CCA security.

Instantiating this conversion also requires providing a NIZK proof system that proves that an HQC.PKE ciphertext is correctly formed. Similarly to [GLOV26b], we plan on building such a

NIZK proof system using MPC-in-the-Head or VOLE-in-the-Head techniques. More precisely, we plan on adapting the techniques used for building the SDitH signature [ABBFG+26; FJR22] to prove low weight of the vectors used in a given HQC.PKE ciphertext.

2.1.4. Expected parameter sets

We expect to produce four parameter sets for each security parameter $\lambda = 128, 192$ and 256 .

NIST-HQC-SSD. This parameter set is identical to the HQC.KEM parameter set. The HQC-SSD decryption matches the KATs for HQC.PKE and HQC-SSD is interchangeable with HQC in this case. However, as presented in [GLOV26a] this parameter set is ill-adapted for thresholdization, as complexity scales in $\mathcal{O}(n_1^{4.5})$, where n_1 is the length of the Reed-Solomon code.

DFR λ -HQC-SSD. This parameter set is made as close to NIST-HQC as possible while still reducing the main performance bottlenecks. There are two possible ways to achieve this: either keep the same families of codes, but find another Reed-Muller Reed-Solomon pair minimizing n_1 or find another family of codes such that the decoding is less complex in the length of the code. The main parameter n (key and ciphertext size) is expected to increase slightly in the trade-off with computational and communication complexity.

SCA-HQC-SSD. This parameter set is optimized for the threshold decryption to be realized in a single computer. This variant could take advantage of the threshold properties to be secure against most Side Channel Attacks (SCA) and Fault Injection Attacks (FIA) in the conservative t -probing model and bit flipping faults model. In this case, the optimisation of parameters does not take into account the number of cycles of communication or the quantity exchanged (all the data being available at all times in memory). This parameter set is mainly motivated as many attacks have been recently published on the decapsulation of HQC.KEM [MNMD25; GMGL24; DGN26; JPLLKK26], and, fault injection attacks on post-quantum schemes flourished in the recent literature [RYBZC23; HPT25; BBK26], targeting a multitude of internal variables of the decryption steps.

DFR0-HQC-SSD. As mentioned earlier, having a perfectly correct scheme may be necessary to apply the Naor-Yung transform to the CPA HQC-SSD. In order to have a DFR of 0, n has to be large enough so that the maximum number of errors is lower than the capacity of the code ($\lfloor \frac{d-1}{2} \rfloor$ for d the minimum distance). The increase of parameters will naturally be detrimental for the performance, but allow stronger security guarantees.

2.2. System Model and Security

Network. We assume the network to be implemented as a synchronous network with a reliable broadcast channel. This broadcast channel ensures that a message sent is the same received by

each party. Each pair of parties has an authenticated and confidential way to communicate, and a trusted setup allows to securely communicate variables and publish the result of a recomposition. This allows for implementing the protocols using secret-sharing based techniques such as Shamir's, in which parties alternately compute locally and exchange data simultaneously in rounds of communications.

UC Security. Our protocols are proven secure using the Universally Composable (UC) framework introduced in [Can01], which compares the behavior of an environment interacting with an adversary either in a “real-world” experiment or an “ideal-world” experiment. In this framework, the decryption protocol is guaranteed to be perfectly correct and to not leak any auxiliary information about the shares of the decryption key or the message, even when used for any amount of time or in parallel.

3. Open-Source Implementation

Code structure. Our implementation will be written in Go, which allows fast, easy to build binaries while being easy to read and having better memory safeguards than the C language (used for the KEM version). If any library is to be used, only open-source libraries will be chosen.

Code progress and availability. The code is still under development. It will be publicly available in a Git repository.

Implementation of the networking model. The network will be simulated using OCI-compliant Linux containers, running them either with the fully open-source tool Podman or the well-known one Docker. Each party will communicate through a server using TCP (even when simulating a broadcast channel). The authenticity, integrity and confidentiality of the link are taken as an hypothesis and not worked on. Further implementation details and choices of protocols will be given with the release of the code.

Testing. As a resulting choice of the networking simulation, the reproducibility of our implementation will be limited by the usual issues encountered using containers to target bitwise and functional reproducibility (host kernel changes, base image availability, temporary files created by the filesystem...). In order to target rebuildability and functional reproducibility under linux-arm64, each version of the dependencies (i.e. base image and packages) will be pinned.

The testing framework will include all malicious behaviour and networking conditions assumed in the security model.

4. Experimental Performance Evaluation

Performance: The client-server architecture will allow us to easily get performance counters that could translate in a realistic environment. However, performance losses are expected

when implementing secure network links (i.e. with integrity, confidentiality and authenticity). Performance counters will include in particular: wall-clock time, number of rounds and the number of bytes sent per round.

5. Licensing, Patent Claims, and Funding

Open-source licenses. The core implementation of the proposed scheme will be released under the LGPL-3.0-only license. Except for the containers' base image (Ubuntu 26.04) and the language compiler, external dependencies will either be open-source (FSF or OSI approved) or will be optional (e.g. Podman can be used instead of Docker to manage the containers).

Patents. To the best of our knowledge, there are no known patents with claims that cover the contents of this submission.

External funding. None.

6. Comparison with Other Submissions

This submission contains the description for a category N2 parameter set, as well as three category S2 parameter sets. In the previous preview phases, there were no category N2 proposals. This submission would be the only one being able to match NIST's PQC standard HQC KATs.

The category S2 cryptosystems from preview phases 1 and 2 are:

- Lattice-based (Amber [[BdLP26](#)])
- Isogeny-based (PQarrots [[AABBB+26](#)])
- Quantum vulnerable (CGGMP [[AABCDGM26](#)], SplitKey [[LPSVM+26](#)])

This submission is, at the time of writing, the only code-based (post-quantum) KEM in the MPTC call.

Scheme	Security	Identifiable abort	Robustness	Computation	Indistinguishability	PQC	Round
NIST-HQC-SSD	Adaptive	Yes	Yes	$O(N)$	?	Yes	Constant
DFRA-HQC-SSD	Adaptive	Yes	Yes	$O(N)$	?	Yes	Constant
SCA-HQC-SSD	Adaptive	Yes	Yes	$O(N)$	?	Yes	Constant
DFR0-HQC-SSD	Adaptive	Yes	Yes	$O(N)$	?	Yes	Constant
Amber-I	Adaptive	?	No	$O(1)$	IND-CCA	Yes	?
Amber-II	Static	?	Yes	$O(T)$	IND-CCA	Yes	?
CGGMP-II	?	Yes	?	?	?	No	?
PQarrot	Static	No	?	$O(T^2)$	?	Yes	Linear
SplitKey (Split-Decryption)	Static	?	?	?	?	No	?

Table 1: Comparative properties of N2 and S2 schemes altogether

References

[AABBB+26] Marius A. Aardal, Shahla Atapoor, Karim Baghery, Andrea Basso, Xavier Bonnetain, Giacomo Borin, Daniele Cozzo, Pierrick Dartois, Luca De Feo, Max Duparc, Jonathan K. Eriksen, Tako Boris Fouotsa, Arthur Herlédan Le Merdy, Riccardo Invernizzi, Samuel Jaques, Yi-Fu Lai, Dania Lazzarini, Jason T. LeGrow, Chloe Martindale, Luciano Maino, Jonas Meers, Michael Meyer, Sikhar Patranabis, Robi Pedersen, Giacomo Pope, Doreen Riepel, Damien Robert, Ryan Rueger, Sina Schaeffler, André Schrottenloher, and Frederik Vercauteren. *PQarrots: Macaw, Kea and Kakapo*. NIST MPTC Preview Writeup. 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/PQarrots-PW01.pdf>.

[AABCDGM26] Michael Adjedj, Tomer Ashur, Amit Singh Bhati, Geoffroy Couteau, Cyprien Delpech de Saint Guilhem, Michael Gutkin, and Nikos Makriyannis. *Distributed ECDSA Signatures*. NIST MPTC Preview Writeup. 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Fireblocks-3MI-c-PW01.pdf>.

[ABBFG+26] Carlos Aguilar Melchor, Slim Bettaleb, Loïc Bidoux, Thibault Feneuil, Philippe Gaborit, Nicolas Gama, Shay Gueron, James Howe, Andreas Hülsing, David Joseph, Antoine Joux, Mukul Kulkarni, Edoardo Persichetti, Tovohery H. Randrianarisoa, Matthieu Rivain, and Dongze Yue. *SDiTH — Syndrome Decoding in the Head*. Tech. rep. available at <https://sdith.org/docs/sdith-v3.0.pdf>. National Institute of Standards and Technology, 2026.

[BBK26] Harrison Banda, Jan Brinkmann, and Juliane Krämer. “Fault Attacks on MPCitH Signature Schemes”. In: *Post-Quantum Cryptography*. Ed. by Magali Bardet and Ruben Niederhagen. Cham: Springer Nature Switzerland, 2026, pp. 409–442. DOI: [10.1007/978-3-032-22698-3_13](https://doi.org/10.1007/978-3-032-22698-3_13). Also at ia.cr/2025/1745.

[BdLP26] Katharina Boudgoust, Rafael del Pino, Oleksandra Lapiha, and Thomas Prest. *Amber: a Family of Efficient Lattice-Based IND-CCA Threshold KEMs*. NIST MPTC Preview Writeup. 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/Amber-PW01.pdf>.

- [BGW88]** Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. “Completeness theorems for non-cryptographic fault-tolerant distributed computation”. In: *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*. STOC '88. Chicago, Illinois, USA: Association for Computing Machinery, 1988, pp. 1–10. DOI: [10.1145/62212.62213](https://doi.org/10.1145/62212.62213). URL: <https://doi.org/10.1145/62212.62213>.
- [Can01]** Ran Canetti. “Universally Composable Security: A New Paradigm for Cryptographic Protocols”. In: *42nd Annual Symposium on Foundations of Computer Science*. Las Vegas, NV, USA: IEEE Computer Society Press, October 2001, pp. 136–145. DOI: [10.1109/SFCS.2001.959888](https://doi.org/10.1109/SFCS.2001.959888). Also at ia.cr/2000/067.
- [DGN26]** Haiyue Dong, Qian Guo, and Denis Nabokov. “Single-Trace Key Recovery Attacks on HQC Using Valid and Invalid Ciphertexts”. In: *Advances in Cryptology – EUROCRYPT 2026*. Ed. by Joan Daemen and Emmanuel Thomé. Cham: Springer Nature Switzerland, 2026, pp. 568–597. DOI: [10.1007/978-3-032-25336-1_20](https://doi.org/10.1007/978-3-032-25336-1_20). Also at ia.cr/2025/1987.
- [FJR22]** Thibault Feneuil, Antoine Joux, and Matthieu Rivain. “Syndrome Decoding in the Head: Shorter Signatures from Zero-Knowledge Proofs”. In: *Advances in Cryptology – CRYPTO 2022, Part II*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Vol. 13508. Lecture Notes in Computer Science. Santa Barbara, CA, USA: Springer, Cham, Switzerland, August 2022, pp. 541–572. DOI: [10.1007/978-3-031-15979-4_19](https://doi.org/10.1007/978-3-031-15979-4_19). Also at ia.cr/2022/188.
- [FO99]** Eiichiro Fujisaki and Tatsuaki Okamoto. “Secure Integration of Asymmetric and Symmetric Encryption Schemes”. In: *Advances in Cryptology — CRYPTO' 99*. Ed. by Michael Wiener. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 537–554. DOI: [10.1007/3-540-48405-1_34](https://doi.org/10.1007/3-540-48405-1_34).
- [GAABB+26]** Philippe Gaborit, Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Edoardo Persichetti, Gilles Zémor, Jurjen Bos, Arnaud Dion, Jerome Lacan, Jean-Marc Robert, Pascal Veron, Paulo Barreto, Shay Gueron, Tim Güneysu, Rafael Misoczki, Nicolas Sendrier, Jean-Pierre Tillich, Valentin Vasseur, Santosh Ghosh, and Jan Richter-Brockmann. *HQC*. Tech. rep. version of October 2026, available at https://pqc-hqc.org/doc/hqc_specifications_2026_10_12.pdf. National Institute of Standards and Technology, 2026.
- [GLOV26a]** Pascal Giorgi, Fabien Laguillaumie, Lucas Ottow, and Damien Vergnaud. *Constant-Round Secure Distributed Decoding and HQC Threshold Decryption*. Cryptology ePrint Archive, Paper 2026/946. 2026. URL: <https://eprint.iacr.org/2026/946>.
- [GLOV26b]** Pascal Giorgi, Fabien Laguillaumie, Lucas Ottow, and Damien Vergnaud. “Threshold Niederreiter: chosen-ciphertext security and improved distributed decoding”. In: *Designs, Codes and Cryptography* 94.3 (February 2026), p. 64. DOI: [10.1007/s10623-025-01795-6](https://doi.org/10.1007/s10623-025-01795-6). Also at ia.cr/2025/757.
- [GMGL24]** Guillaume Goy, Julien Maillard, Philippe Gaborit, and Antoine Loiseau. “Single trace HQC shared key recovery with SASCA”. In: *IACR Transactions on Cryptographic Hardware and Embedded Systems* 2024.2 (March 2024), pp. 64–87. DOI: [10.46586/tches.v2024.i2.64-87](https://doi.org/10.46586/tches.v2024.i2.64-87). URL: <https://tches.iacr.org/index.php/TCHES/article/view/11421>. Also at ia.cr/2023/1590.

[HPT25] Calvin Abou Haidar, Quentin Payet, and Mehdi Tibouchi. “Crowhammer: Full Key Recovery Attack on Falcon with a Single Rowhammer Bit Flip”. In: *Advances in Cryptology – CRYPTO 2025*. Ed. by Yael Tauman Kalai and Seny F. Kamara. Cham: Springer Nature Switzerland, 2025, pp. 103–135. DOI: [10.1007/978-3-032-01901-1_4](https://doi.org/10.1007/978-3-032-01901-1_4). Also at ia.cr/2025/1042.

[JPLKK26] Jaeho Jeon, Yongseong Park, Jaeyeon Lee, Suseong Lee, Donghyun Kim, and Young-Sik Kim. *Codeword Masking Can Be Harmful Under Replay Attacks on HQC*. Cryptology ePrint Archive, Paper 2026/071. 2026. URL: <https://eprint.iacr.org/2026/071>.

[Lin03] Yehuda Lindell. “A Simpler Construction of CCA2-Secure Public-Key Encryption under General Assumptions”. In: *Advances in Cryptology – EUROCRYPT 2003*. Ed. by Eli Biham. Vol. 2656. Lecture Notes in Computer Science. Warsaw, Poland: Springer Berlin Heidelberg, Germany, May 2003, pp. 241–254. DOI: [10.1007/3-540-39200-9_15](https://doi.org/10.1007/3-540-39200-9_15). Also at ia.cr/2002/057.

[LPSVM+26] Peeter Laud, Alisa Pankova, Nikita Snetkov, Jelizaveta Vakarjuk, Petr Muzikant, Aivo Kalu, Burak Can Kus, Semjon/Sona Kravtšenko, Raul-Martin Rebane, and Mart Oruaas. *SplitKey*. NIST MPTC Preview Writeup. 2026. URL: <https://csrc.nist.gov/csrc/media/Projects/threshold-cryptography/documents/TCall-1/SplitForge-PW01.pdf>.

[MNMD25] Nathan Maillet, Cyrius Nugier, Vincent Migliore, and Jean-Christophe Deneuville. “Key Recovery from Side-Channel Power Analysis Attacks on Non-SIMD HQC Decryption”. In: *Advances in Cryptology – CRYPTO 2025*. Ed. by Yael Tauman Kalai and Seny F. Kamara. Cham: Springer Nature Switzerland, 2025, pp. 70–102. DOI: [10.1007/978-3-032-01901-1_3](https://doi.org/10.1007/978-3-032-01901-1_3). Also at ia.cr/2025/1270.

[NY90] Moni Naor and Moti Yung. “Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks”. In: *22nd Annual ACM Symposium on Theory of Computing*. Baltimore, MD, USA: ACM Press, May 1990, pp. 427–437. DOI: [10.1145/100216.100273](https://doi.org/10.1145/100216.100273).

[RYBZC23] Prasanna Ravi, Bolin Yang, Shivam Bhasin, Fan Zhang, and Anupam Chattopadhyay. “Fiddling the Twiddle Constants - Fault Injection Analysis of the Number Theoretic Transform”. In: *IACR Transactions on Cryptographic Hardware and Embedded Systems 2023.2* (March 2023), pp. 447–481. DOI: [10.46586/tches.v2023.i2.447-481](https://doi.org/10.46586/tches.v2023.i2.447-481). URL: <https://tches.iacr.org/index.php/TCHES/article/view/10290>. Also at ia.cr/2022/824.

[NIST-IR8214C] Luís T. A. N. Brandão and René Peralta. *NIST First Call for Multi-Party Threshold Schemes*. (National Institute of Standards and Technology) NIST Internal Report (NISTIR) 8214C. 2026. DOI: [10.6028/NIST.IR.8214C](https://doi.org/10.6028/NIST.IR.8214C).