White paper:

Quantum Computing Threat: The First NIST Post-Quantum Cryptographic Standards

PQShield

Updated August 2022

Contents

Executive Summary

The NIST standardization process

In 2016, the National Institute of Standards and Technology (NIST) launched a open and world-wide effort to propose, analyze and eventually standardize post-quantum cryptographic schemes. The two primitives covered by this standardization process were:

- signature schemes;
- key-establishment schemes.

In July 2022, more than 5 years after its initial call for proposals, NIST announced the first results of its standardization process [NIS22]. NIST's decision means that:

1. an initial selection of schemes will be standardized;
2. this initial selection will be completed in the future with additional schemes.

Selected standards

For key-establishment, NIST has selected one unique scheme: Kyber (page 22). The mathematical object underlying the security of Kyber is the so-called class of structured lattices.

For signatures, NIST has selected three schemes. The *primary* standard is Dilithium (page 12), whereas the *secondary* standards for specific applications are Falcon (page 13) and SPHINCS$^+$ (page 14). Dilithium and Falcon are based on structured lattices, whereas SPHINCS$^+$ is based on hash functions (such as SHA-2 or SHA-3).

The next steps

Kyber, Dilithium, Falcon and SPHINCS$^+$ will be standardized. NIST plans to deliver the first set of specifications in 2024. Meanwhile, NIST intends to diversify its portfolio by standardizing schemes that do not rely on structured lattices. This will be done via two processes:

1. A selection of four key-establishment schemes will continue to be scrutinized by NIST and the community: BIKE (page 23), Classic McEliece (page 24), HQC (page 25) and SIKE (page 26). None of these schemes rely on structured lattices.
2. NIST will open a new call for additional signature schemes, with a submission deadline expected to be 2023. The explicit goal is to have signature schemes that do not rely on structured lattices, and/or signature schemes with small signatures and short verification.

In both cases, the process may lead to the standardization of one or more schemes.

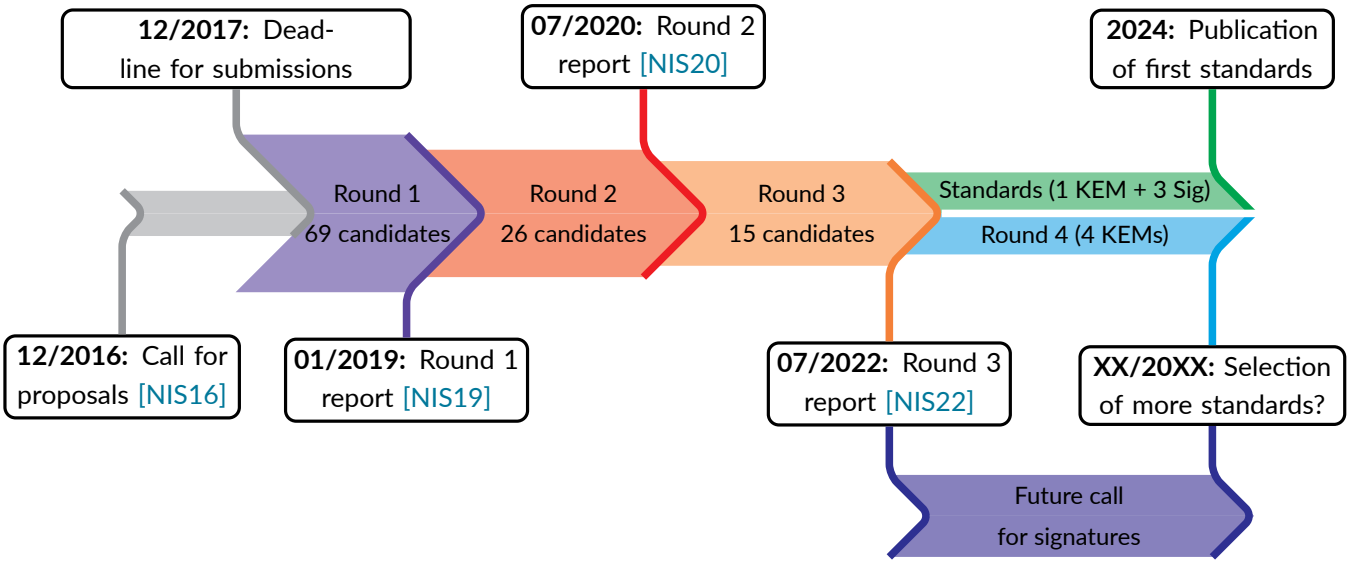
1 The NIST Standardization Process

A question that may come to mind regarding standardization of post-quantum cryptography is:

Why did NIST decide to standardize post-quantum cryptography now
if quantum computers are not yet in practical use?

The reason is simple: standardizing and deploying new technology takes time. For example, the hash function SHA-2 has been standardized since 2001 to replace SHA-1; yet the latter can still be found in many places[1], despite several practical attacks against its collision resistance [SBK+17, LP19, LP20]. On the other hand, quantum computing is a fast-moving field, attracting hundreds of millions of dollars[2] in yearly funding. In this context, an early standardization by NIST gives organizations more time and flexibility to carry out a smooth transition to quantum-safe cryptography.

There are a number of standardization efforts currently underway (by ETSI in Europe, CACR in China, etc.), but we focus on the one by NIST since it is by far the most documented and has attracted a significant amount of industrial and academic attention. NIST's post-quantum standardization process was announced in 2016 [NIS16], with the goal to standardize post-quantum signature schemes and key-establishment schemes.



82 submissions were filed in November 2017, of which 69 were considered "complete and proper" as per NIST's minimal acceptance criteria and selected as Round 1 candidates (49 for key-establishment, 20 for signatures). In January 2019 [NIS19], NIST selected 26 schemes as Round 2 candidates (17 for key-establishment, 9 for signatures). In July 2020 [NIS20], 15 schemes were selected as Round 3 candidates. Finally, in July 2022 [NIS22], NIST decided to:

- Standardize one key-establishment scheme (more precisely a *key encapsulation mechanism*, or KEM) and three signature schemes (first standards expected in 2024);
- Select four KEMs for further study in the so-called *Round 4*;
- Open a future call to diversify its signature portfolio (submission deadline expected in 2023).

[1] See for example The Github Blog: *Highlights from Git 2.29*.

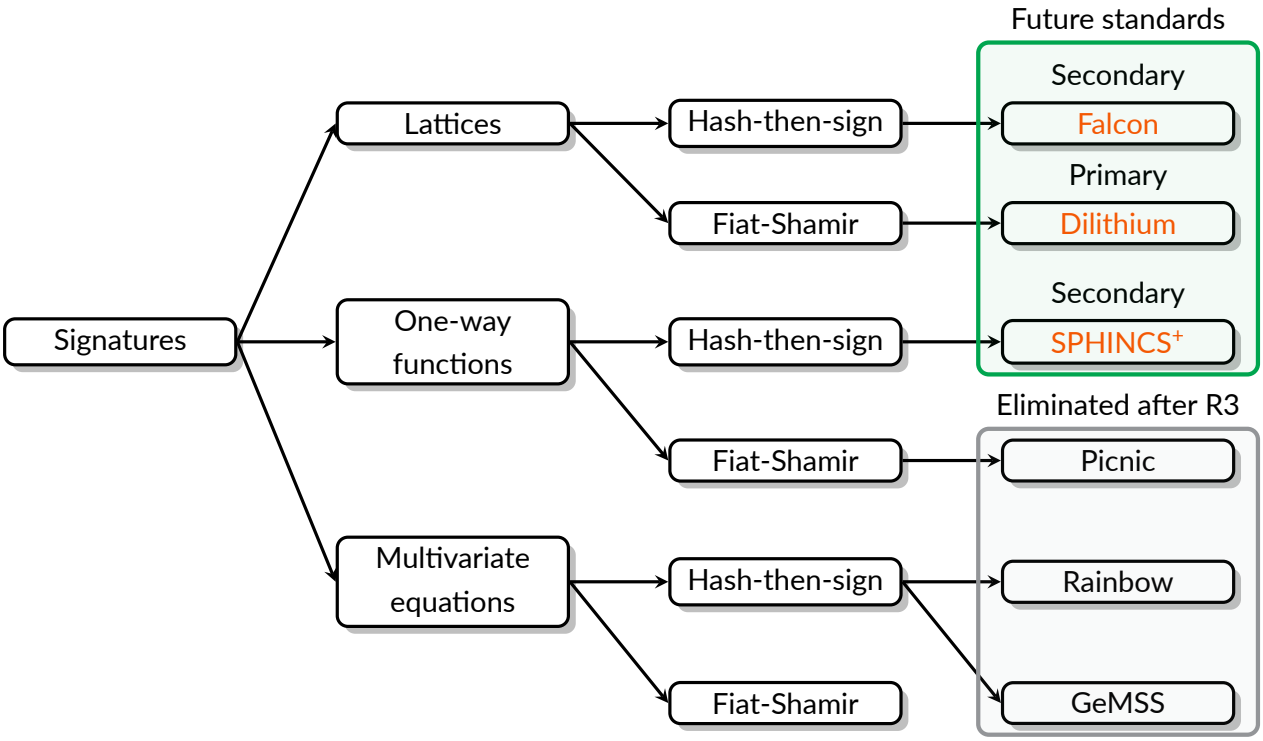
[2] Nature: *Quantum gold rush: the private funding pouring into quantum start-ups*

2 Signature Schemes

We first discuss signature schemes that will be standardized. In Round 1 of this process (December 2017), 20 digital signature schemes were accepted [NIS17]. After a preliminary analysis by the cryptographic community, NIST selected 9 of these 20 schemes for Round 2 of the standardization process [NIS19]. In July 2020, NIST narrowed the selection to 6 schemes [NIS20]. In July 2022, NIST announced its decision [NIS22] to select three standards:

- Dilithium (primary standard);
- Falcon (secondary standard);
- SPHINCS$^+$ (secondary standard).

Dilithium and Falcon are based on hardness assumptions about structured lattices, and SPHINCS$^+$ is based on hardness assumptions about hash functions. Although there exist code-based or isogeny-based signature schemes, none are in this shortlist (because they were either eliminated at Round 1, or proposed after the submission deadline) so have not been included here. Schemes based on multivariate polynomials were eliminated at the end of Round 3. An overview of the standard signature schemes can be found in the figure below.



In page 6, we briefly present the schemes selected by NIST for standardization. In addition, we provide a comparative performance study of the 3 selected standards in pages 7 to 10. Finally, for each scheme, one page summarizes its main properties (pages 12 to 14).

Selected Standards and Next Steps

Primary standard: Dilithium

NIST has decided to select Dilithium (page 12) as its primary standard for signature.

This choice is thoroughly explained in [NIS20]. Dilithium and Falcon are both secure, efficient schemes. However, Dilithium is more balanced overall: its key generation and signing procedures are faster, it is simpler to implement and doesn't have unusual requirements such as the need for floating-point arithmetic support (which Falcon does).

In addition, several embedded implementation for Dilithium have been proposed [LSG21, BNG21, GKS21, RMJ+21, BNG22], which is a testament to its ability to be deployed in constrained environments. For all these reasons, Dilithium has been selected as the primary standard for signatures.

Secondary standard: Falcon

In [NIS20], NIST stated that they would standardize either Dilithium or Falcon, but not both. It has therefore been a relative surprise that both schemes have been standardized: Dilithium is selected as a primary standard, and Falcon (page 13) is selected as a secondary standard.

NIST explained this choice in [NIS22]. Falcon has much smaller bandwidth requirements, which according to various third parties makes it a potentially better choice than Dilithium for specific use-cases such as TLS with native floating-point support [SKD20] and V2V communications [BMTR21].

Secondary standard: SPHINCS+

SPHINCS+ (page 14) has also been selected as a secondary standard.

SPHINCS+ relies purely on assumptions based on hash functions. These assumptions are perceived as much more conservative than the structure lattice assumptions that Dilithium and Falcon rely on. The main downside of SPHINCS+ is that its performances are much worse than for the two other standards: for example, the signature size, verification time and signing time are respectively one, two and three orders of magnitude higher than for, say, Dilithium.

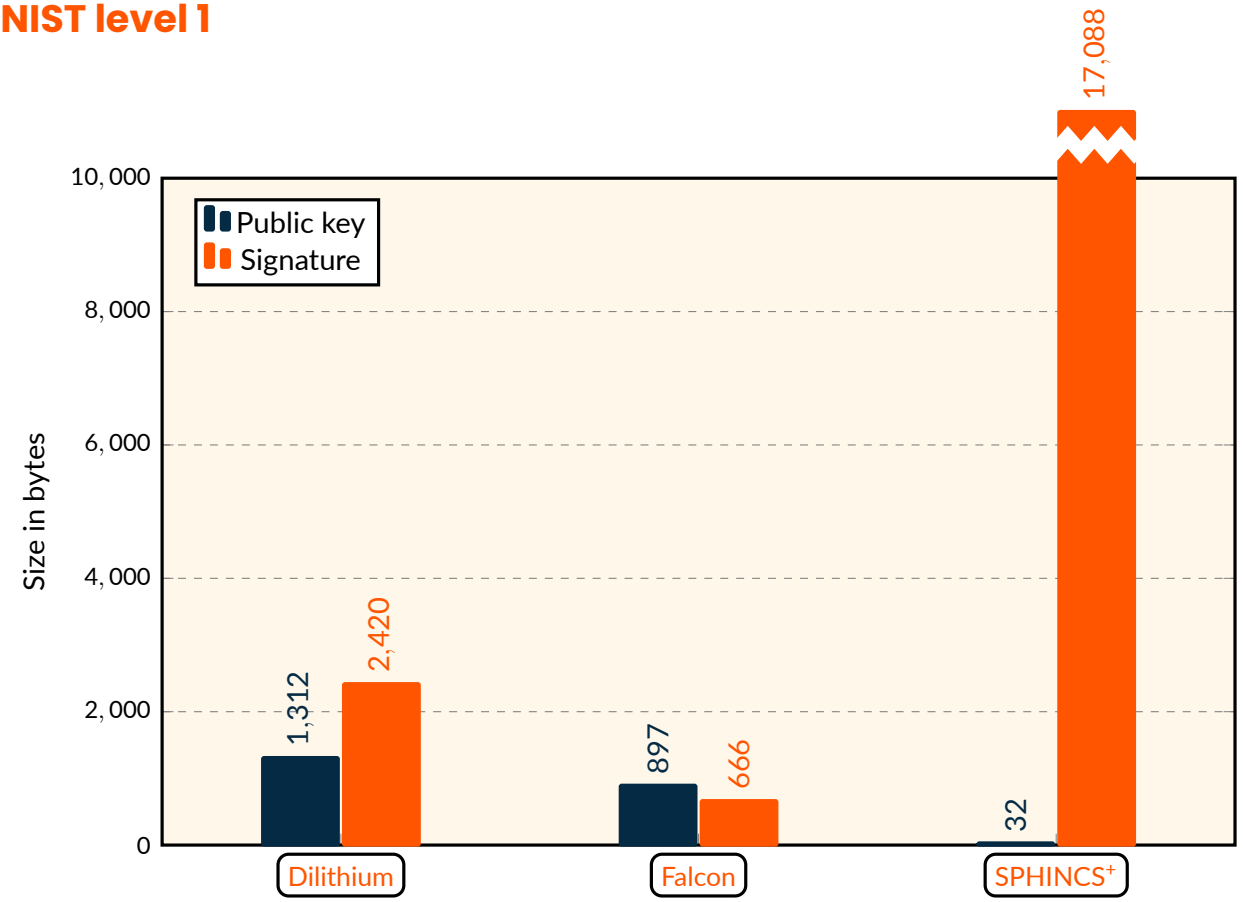
Future call for signatures

In [NIS22], NIST has announced its intent to open a new call for proposals for signatures in the future. One of the stated goals is to diversify its portfolio, by relying on other assumptions than structured lattices. Another potential goal is to standardize signatures with small signatures and/or fast verification. NIST expect the deadline for submissions to be in 2023, see §5 in [NIS22] for more details.

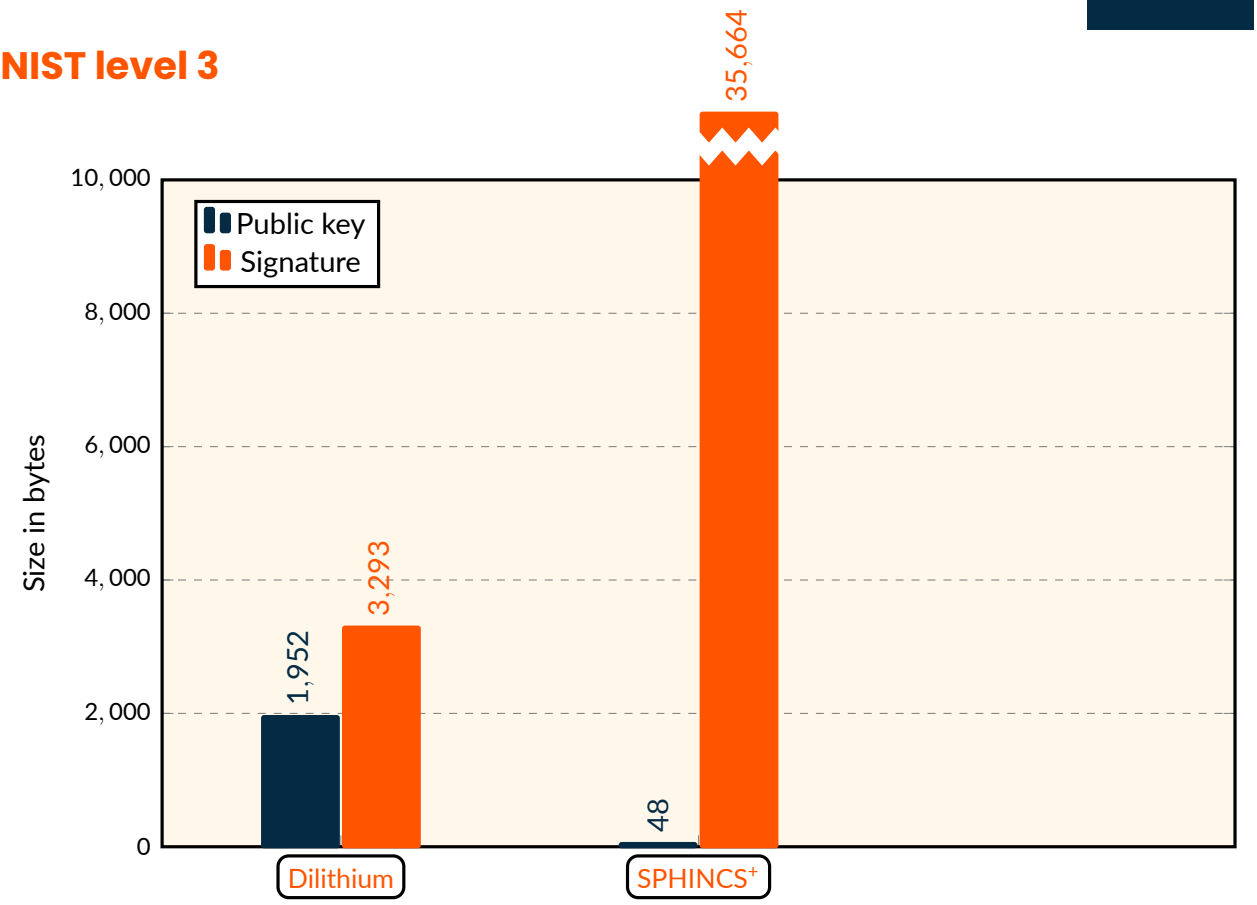
Communication Costs

We now provide a detailed comparison of the communication costs of the 3 selected standards for three security levels: NIST Level 1, 3 and 5 (conjectured at least as secure as AES-128, AES-192 and AES-256, respectively). While lattice-based schemes Dilithium and Falcon have small public keys and small signatures, hash-based SPHINCS$^+$ has tiny public keys but large signatures.

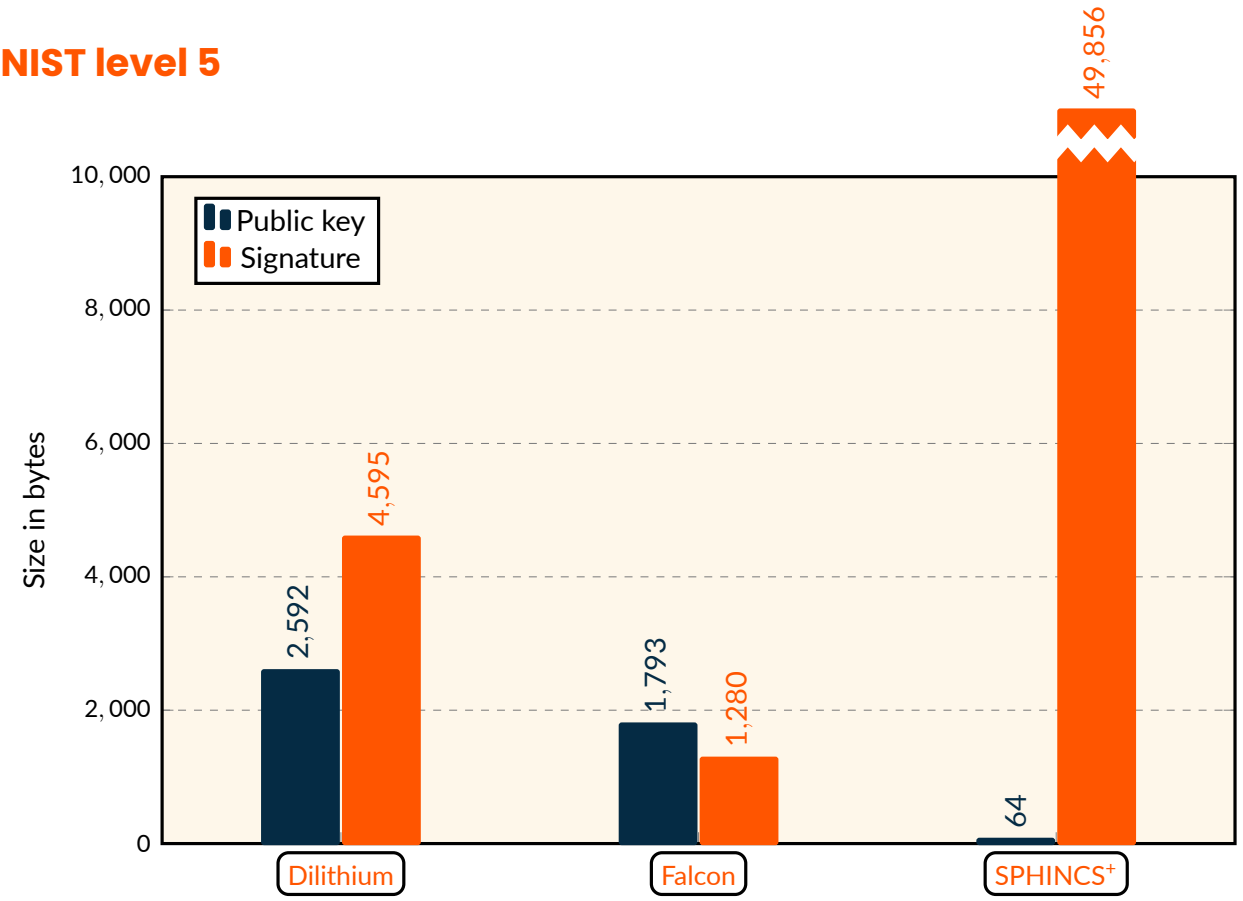
NIST level 1



NIST level 3



NIST level 5

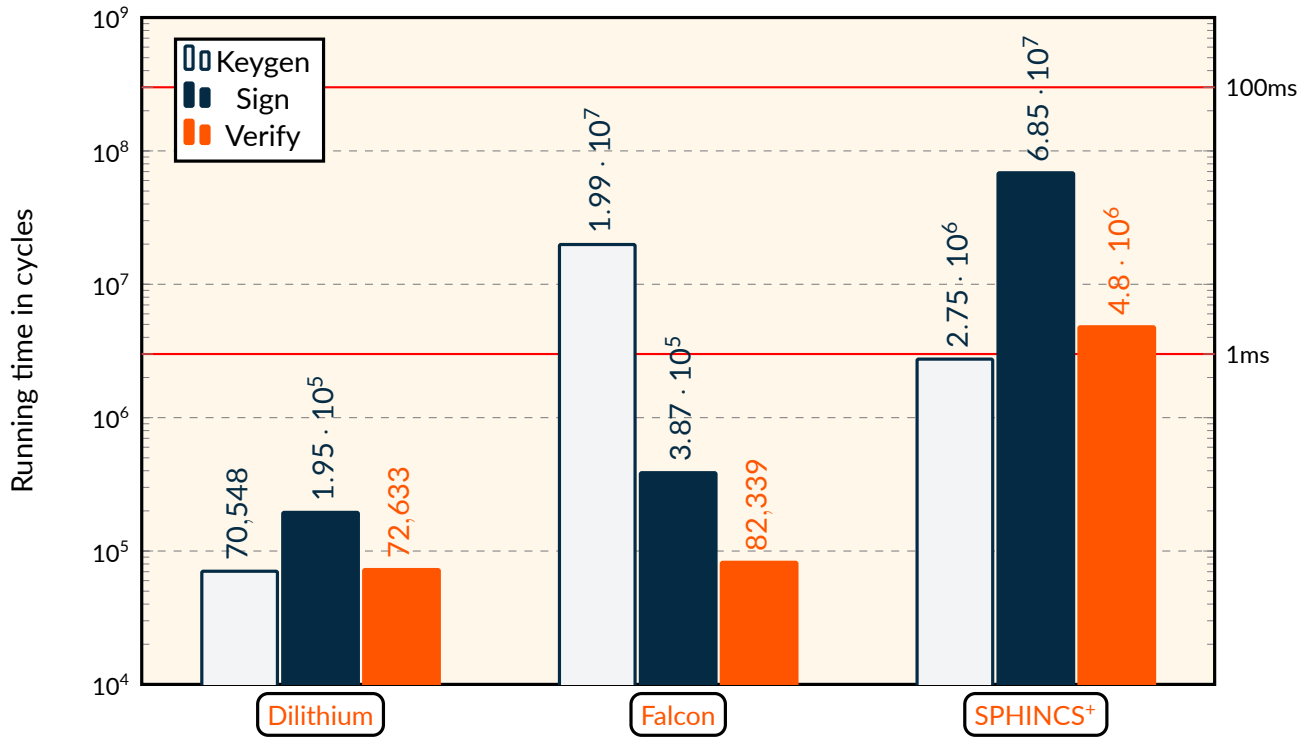


Computational Costs

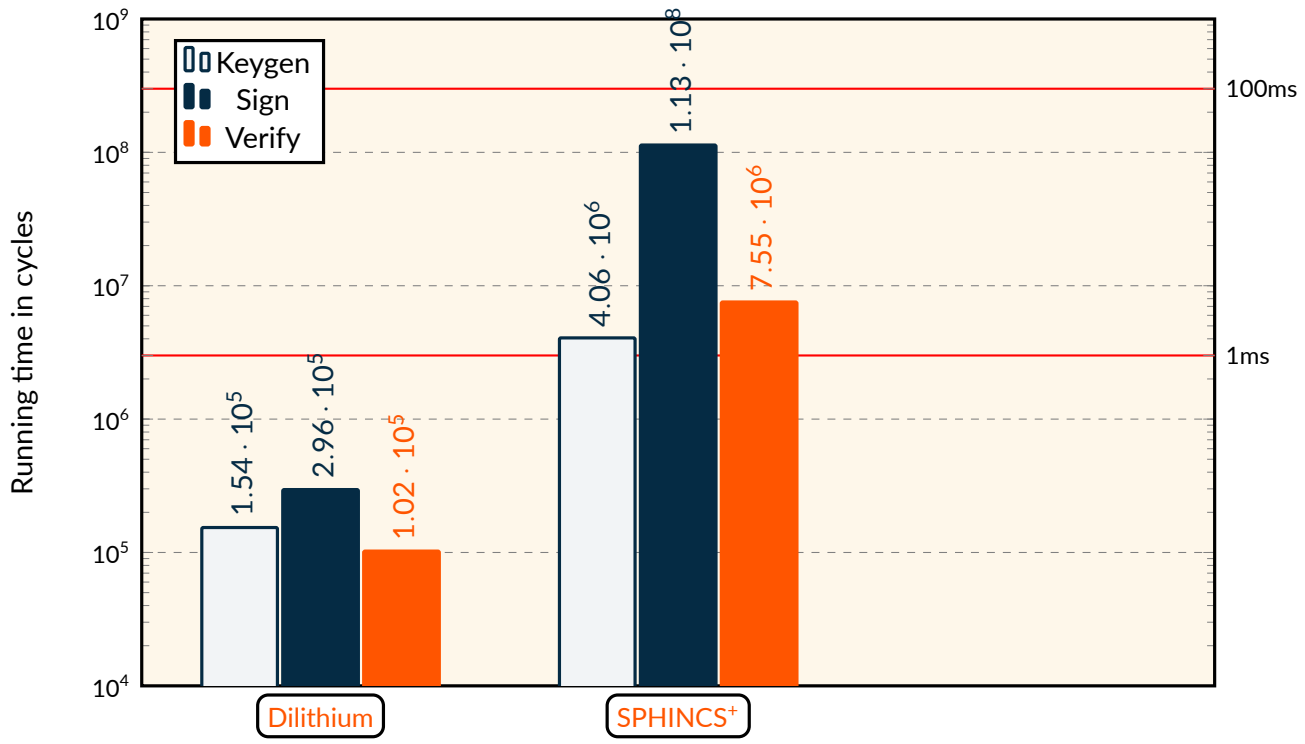
We now compare the running times in cycles of the 3 selected standards, for optimized implementations targeting x64 platforms. All numbers are extracted from the specification documents of the schemes (which might be inaccurate) and were obtained on different platforms. Therefore, they may not enable a completely fair comparison. To make these numbers less abstract, each graph also contains two horizontal red lines that correspond respectively to 1 millisecond and 1 second on a microprocessor with a clock frequency of 3GHz, which is typical for microprocessors in personal computers.

We observe a high disparity between candidates. For example, the overall fastest signature scheme at the highest security level (Dilithium) has key generation, signing and verification procedures that are, respectively, about 140, 1200 and 100 times faster than the overall slowest one (SPHINCS$^+$). Note that raw performances do not tell the full story, since SPHINCS$^+$ relies on what appear to be more conservative assumptions than any other signature scheme presented here.

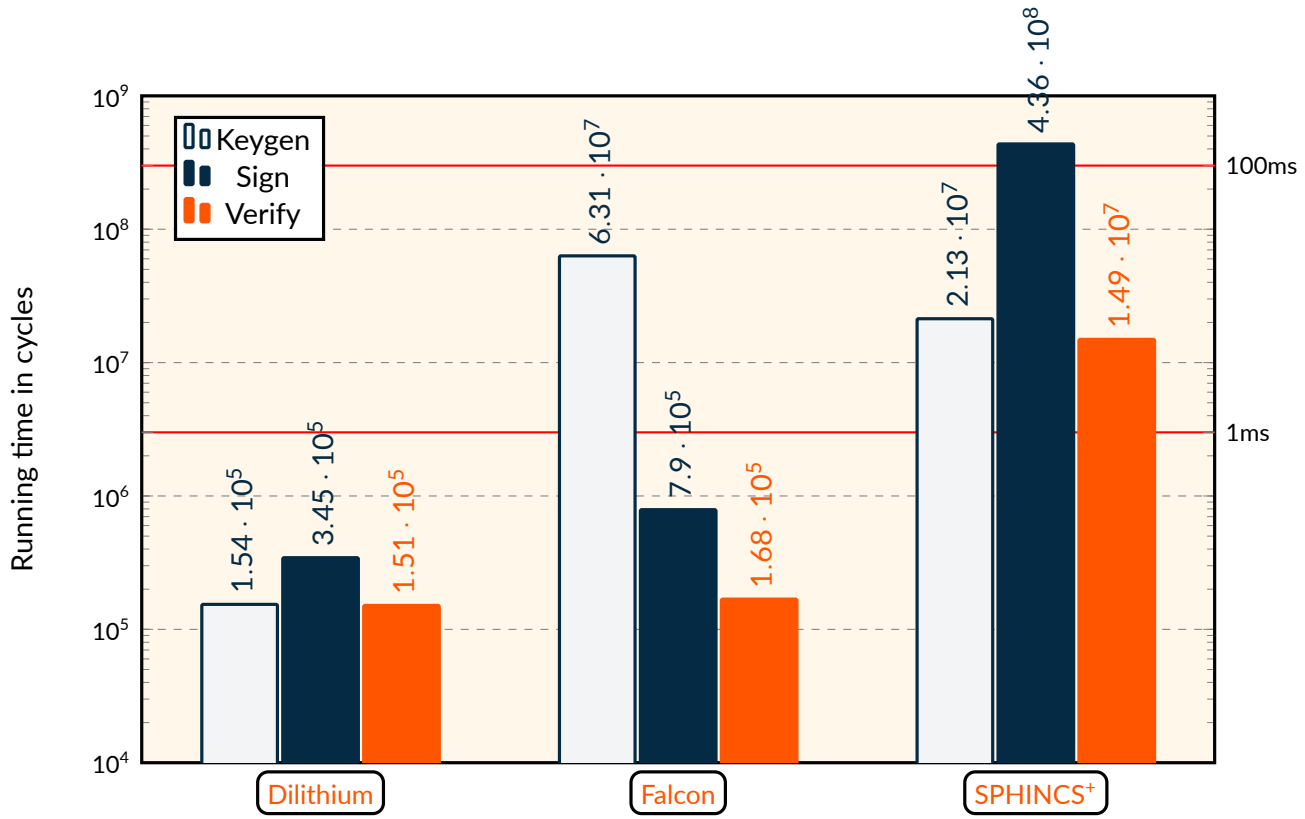
NIST level 1



NIST level 3



NIST level 5



Breakdown of Each Scheme

For each signature scheme, we now provide the following information:

- **The paradigm** can be either Hash-then-sign or Fiat-Shamir. Even in the same family, two schemes based on different paradigms often end up with very different properties.
- **The family** can be either Lattices or Hash functions.
- **The underlying hard problem(s)** is specified.
- The **symmetric primitives** and the **type of randomness** used are specified. These are not always important theoretically, but can have a huge impact on performance. For example, SPHINCS$^+$ is very dependent on the underlying symmetric primitive, and Gaussian distributions (used in Falcon) can be hard to generate in a masked fashion.
- Links to **the specification**, **the website** (if any) and to **related works** are also provided.
- A **short summary** highlights the key facts about the scheme.
- Finally, a **performance table** is provided.

3 Dilithium (Primary standard)

Type: Signature
Paradigm: Fiat-Shamir
Family: Lattices
Hard Problems: Module-LWE (Learning With Errors), Module-SIS (Short Integer Solution)
Sym. primitives: SHAKE, AES
Randomness: Uniform, and uniform over the set $\mathcal{B}_\tau$ of ternary vectors with L_1 norm τ
Specification: [LDK+20]
Website: `https://pq-crystals.org/dilithium/`
Related Works: [Lyu09, Lyu12, GLP12, DDLL13, BG14, KLS18, DKL+18, BP18b]

NIST's overall assessment [NIS22]

"Dilithium is a signature scheme with high efficiency, relatively simple implementation, a strong theoretical security basis, and an encouraging cryptanalytic history. It is an excellent choice for a broad range of cryptographic applications and is, thus, the primary signature algorithm selected by NIST for standardization at this time."

Design rationale and physical attacks

Dilithium is based on the Fiat-Shamir with Aborts paradigm [Lyu09]. It implements two notable tricks: the first one, introduced in [GLP12], divides the size of the public key almost in half. A related trick by [BG14] reduces the size of the signature by half, by sending one ring element instead of two. It also borrows ideas from BLISS [DDLL13].

The design of Dilithium has been heavily influenced by the numerous side-channel attacks to which its prececessor, BLISS, has been subjected [BHLY16, PBY17, EFGT17, BDE+18]. To thwart these attacks, Dilithium uses uniform distributions instead instead of BLISS's Gaussians. A masked implementation of Dilithium has been proposed in [MGTF19].

Underlying assumptions

Dilithium relies on the (decisional) Module-LWE and Module-SIS problems [LS15]. In addition, the security proof in the QROM relies on a new problem called SelfTargetMSIS [KLS18]. New results suggest that this problem might not be necessary after all, see next paragraph.

Security model

In the ROM, Dilithium is claimed to be SEU-CMA under the (decisional) Module-LWE and Module-SIS problems; SEU-CMA stands for the classical notion of *Strong Existential Unforgeability under Chosen-Message Attack*. In the QROM, it is claimed to be SEU-CMA under Module-LWE, Module-SIS and SelfTargetMSIS. New results [DFMS19, LZ19] indicate that the hypothesis SelfTargetMSIS may not be necessary after all.

Embedded implementations

Several implementation of Dilithium on embedded devices have been proposed, for example on FPGAs [LSG21, BNG21, RMJ+21, BNG22] or ARM Cortex micro-controllers [GKS21].

NIST level	|SK| (bytes)	|PK| (bytes)	|sig| (bytes)	KG (cycles)	Sign (cycles)	Verify (cycles)
2	-	1312	2420	70548	194892	72633
3	-	1952	3293	153856	296201	102396
5	-	2592	4595	153936	344578	151066

4 Falcon (Secondary standard)

Type: Signature
Paradigm: Hash-then-sign
Family: Lattices
Hard Problems: NTRU
Sym. primitives: SHAKE-256
Randomness: Noncentered discrete Gaussians
Specification: [PFH+20]
Website: `https://falcon-sign.info/`
Related Works: [HHP+03, GPV08, SS13, DLP14, DP16, OSHG19]

NIST's overall assessment [NIS22]

"*Falcon was chosen for standardization because NIST has confidence in its security (under the assumption that it is correctly implemented) and because its small bandwidth may be necessary in certain applications.*"

Design

Falcon is based on the GPV framework [GPV08] for obtaining hash-then-sign schemes over lattices. As first suggested by [SS13, DLP14], the design is instantiated over the very compact class of NTRU lattices [HHP+03] in order to minimize the bandwidth cost. Falcon is the Round 3 signature with the smallest communication cost (public key + signature).

Algorithmic optimisations

Falcon exploits the algebraic structure of cyclotomic rings in order to optimize its efficiency, notably via the use of a *Fast Fourier Sampling* algorithm [DP16] in the signing procedure, and of a *tower-of-rings* algorithm [PP19] during key generation. Both algorithms yields a $\tilde{O}(n)$-factor improvement compared to previous algorithms, n being the degree of the base ring $\mathbb{Z}[x]/(x^n + 1)$.

Variants

A few variants of Falcon have been proposed, such as a module version, ModFalcon [CPS+20], or a masking-friendly version, Mitaka [EFG+22]. In addition, a ring signature variant has been proposed, Raptor [LAZ19].

Implementation

Falcon uses floating-point arithmetic (FPA), which can make its implementation delicate on platforms that don't support FPA natively. In this case, FPA needs to be emulated. [OSHG19, Por19] have proposed implementations of Falcon on ARM Cortex-M4; both use memory-laziness tricks in order to reduce its memory footprint.

Physical attacks

Recently, two side-channel attacks against unprotected implementations of Falcon have been proposed. The first one [KA21, GMRR22] targets floating-point multiplications, the second one [GMRR22] is a variation of the hidden parallepiped attack [NR06].

NIST level	|SK| (bytes)	|PK| (bytes)	|sig| (bytes)	KG (cycles)	Sign (cycles)	Verify (cycles)
1	-	897	666	19872000	386678	82339
3	-	-	-	-	-	-
5	-	1793	1280	63135000	789564	168498

5 SPHINCS$^+$ (Secondary standard)

Type: Signature
Paradigm: Hash-then-sign
Family: Hash functions
Hard Problems: Multi-target second-preimage resistance of a hash function family
Sym. primitives: SHAKE-256, SHA-256 or Haraka (underlying hash function)
Randomness: Uniform
Specification: [HBD+20]
Website: `https://sphincs.org/`
Related Works: [BDH11, Hül13, BHH+15, HRS16, AE17, AE18]

NIST's overall assessment [NIS22]

"SPHINCS$^+$ was selected for standardization because it provides a workable (albeit rather large and slow) signature scheme whose security seems quite solid and is based on an entirely different set of assumptions than those of our other signature schemes to be standardized.
The two attacks related to SHA-256-based parameters claiming category 5 security will need to be carefully considered when selecting which parameters of SPHINCS$^+$ to standardize."

Design rationale and optimizations

SPHINCS$^+$ is a stateless hash-based signature scheme. It follows the framework introduced in [BHH+15], which combines Merkle trees, Goldreich trees and hash-based few-times signatures (or FTS). SPHINCS$^+$ introduces a few optimizations such as the use of tweakable hash functions [HRS16] against multi-target attacks. HORST, an FTS used in [BHH+15], has been replaced by FORS, a more secure FTS which also provides smaller signatures. See also [BHK+19].

Variants

SPHINCS$^+$ admits several variants: there are 3 security levels (128, 192 or 256) and 3 choices for the underlying building block (SHAKE-256, SHA-256 or Haraka). Additionally, there is a "small"/"fast" distinction (smaller signatures vs faster signing), as well as a "simple"/"robust" distrinction (simpler, faster scheme vs more conservative security argument). Hence there are $3 \times 3 \times 2 \times 2 = 36$ variants. The performance numbers provided here are for SPHINCS$^+$-SHA-256-fast-robust.

Security proof?

While some simple hash-based signatures have security reductions to standard assumptions over generic hash functions, SPHINCS$^+$ is one of the more complex schemes in this family, and no security proof is known for it (yet). See also [BH19].

Attacks and physical attacks

During Round 3, it was pointed out in the NIST mailing list [aut20] that SPHINCS$^+$ might not reach its claimed security levels for the so-called NIST Level 5, due to the internal structure of SHA-256.
A side-channel attack [KGB+18] has shown how an unprotected implementation can leak part of the private key. Similarly, [CMP18, GKPM18] showed how to recover the private key via fault injection.

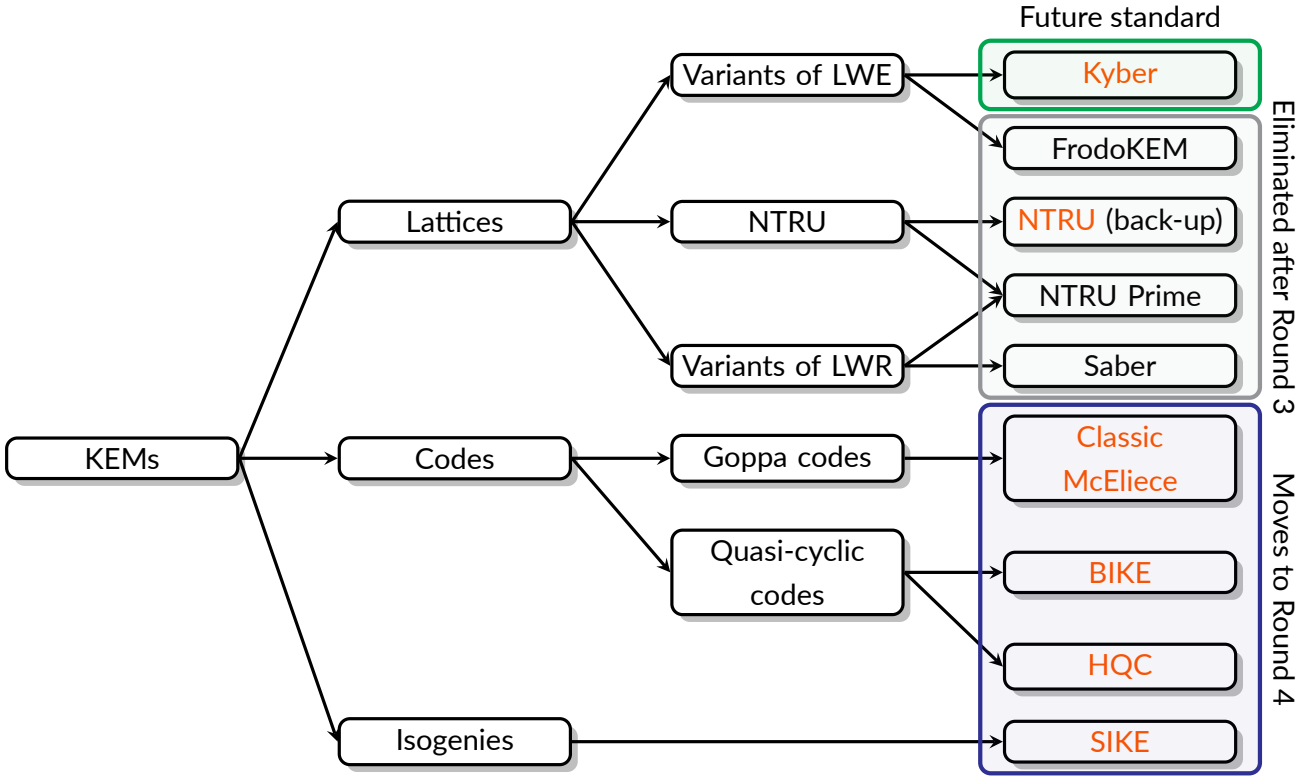
NIST level	|SK| (bytes)	|PK| (bytes)	|sig| (bytes)	KG (cycles)	Sign (cycles)	Verify (cycles)
1	64	32	17088	2748026	68541826	4801338
3	96	48	35664	4063066	113484456	7552358
5	128	64	49856	21327470	435984168	14938510

6 Key-Encapsulation Mechanisms

We now discuss the key-establishment schemes. In Round 1 of NIST's standardization process (December 2017), 49 submissions for key-establishment were accepted [NIS17]. After a preliminary analysis by the cryptographic community, NIST selected 17 of these 49 submissions for Round 2 [NIS19]. Round 3 of the standardization process [NIS20] narrowed this selection to 9 schemes. In July 2022, NIST announced its decision [NIS22]:

- Kyber *will* be standardized;
- Four schemes (BIKE, Classic McEliece, HQC, SIKE) will be selected for further study during Round 4 of the standardization process, at the end of which one or more schemes *might* be standardized;
- NTRU is kept as a back-up for standardization. If patent negotiations around patents related to Kyber fail, NIST *might* standardize NTRU instead of Kyber.

These submissions are based on three families of hardness assumptions: codes, lattices or isogenies. Candidates based on multivariate equations were eliminated at Round 1. Some submissions also propose an encryption scheme or a key-exchange protocol, but all submissions propose a Key Encapsulation Mechanism (henceforth KEM). This KEM is typically obtained by applying to a base key-exchange/encryption scheme a *CCA transform* which provides increased security guarantees against active attackers. Thus, for simplicity we will only consider KEMs. An overview of the KEMs can be found in the figure below.



In page 16, we briefly present the schemes selected by NIST for immediate standardization or further study. In addition, we provide a comparative performance study of all schemes in pages 17 to 20. Finally, for each scheme, one page summarizes its main properties (pages 22 to 27).

Selected Standard and Next Steps

Primary standard: Kyber

As of the July 2022 announcement [NIS22], NIST has standardized a single KEM: Kyber (page 22). NIST made this choice because of the solid performance profile and well-studied underlying security assumptions of Kyber.

Note that four KEMs based on structured lattices (Kyber, Saber, NTRU, NTRU Prime) made it to Round 3, and their performances and security assumptions were comparable. In the end, Kyber has been selected due to small advantages such as (a) a faster key generation and (b) a security assumption perceived as better understood.

Note that three patents have been publicly discussed as potentially covering Kyber: [EP2537284B1], [US9246675B2], and [US9698986B1]. In the perspective of standardizing Kyber, NIST has entered negotiations with owners of these patents.

Selected for further study: BIKE, Classic McEliece, HQC, SIKE

NIST has expressed its desire to diversify its portfolio by standardizing schemes not based on structure lattices (as is Kyber). For this reason, [NIS22] has announced that four schemes have been selected for further study in the Round 4 of the NIST PQC standardization process:

- BIKE (page 23), based on quasi-cyclic codes;
- Classic McEliece (page 24), based on Goppa codes;
- HQC (page 25), based on quasi-cyclic codes;
- SIKE (page 26), based on supersingular isogenies.

The standardization process continues for these schemes. None of them is standardized at the moment, but NIST *might* decide to standardize one or more of them in the future.

Back-up: NTRU

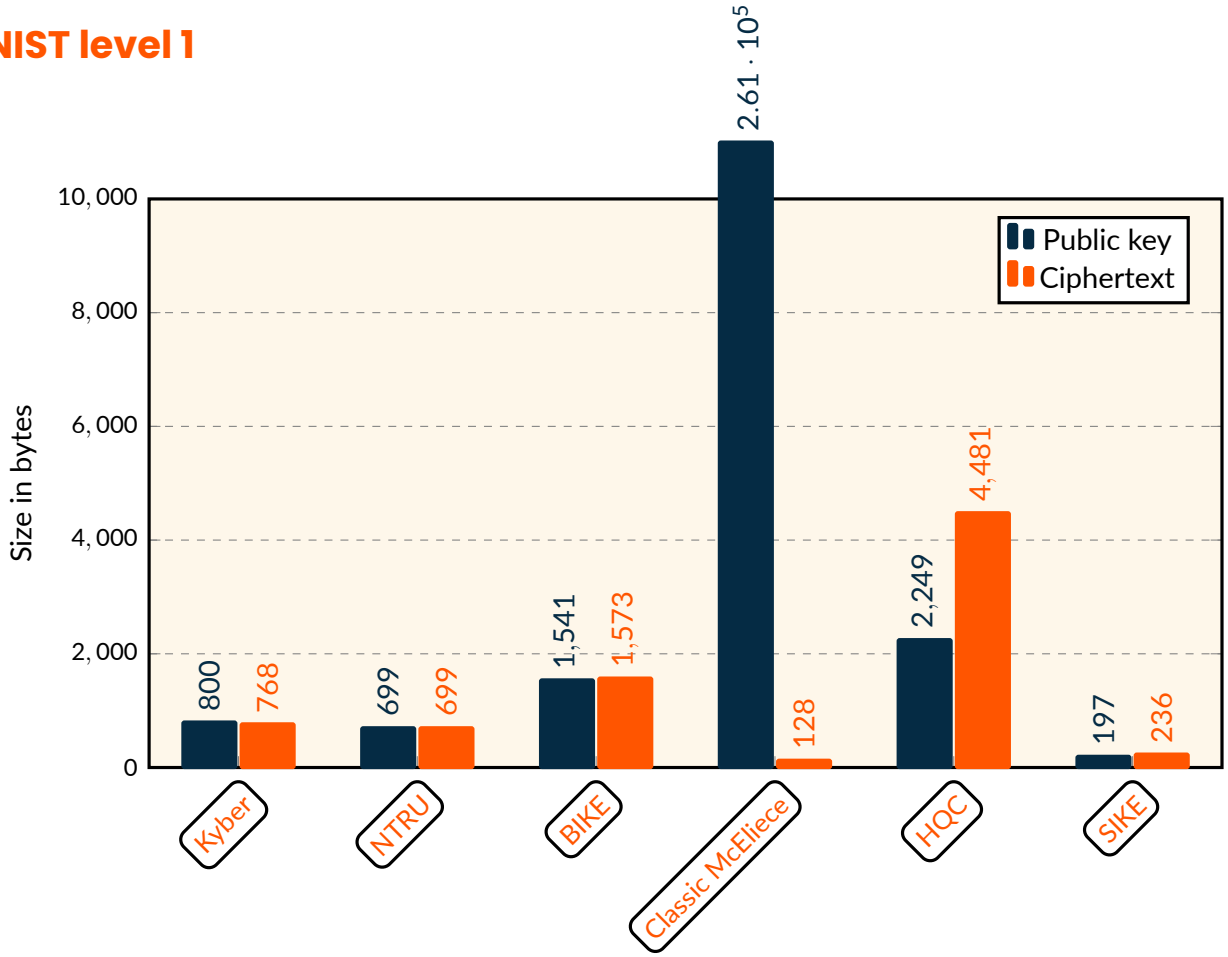
The case of NTRU (page 27) is unusual. NTRU has been *eliminated* in NIST's July 2022 announcement [NIS22]. However, NIST has stated in Footnote 6 of [NIS22] that if agreements with patent owners (see above) are not executed by the end of 2022, NIST may consider standardizing NTRU instead of Kyber.

This indicates that NTRU is considered by NIST as a back-up for standardization in case patent negotiations related to Kyber are not resolved. Hence we include NTRU in this document.

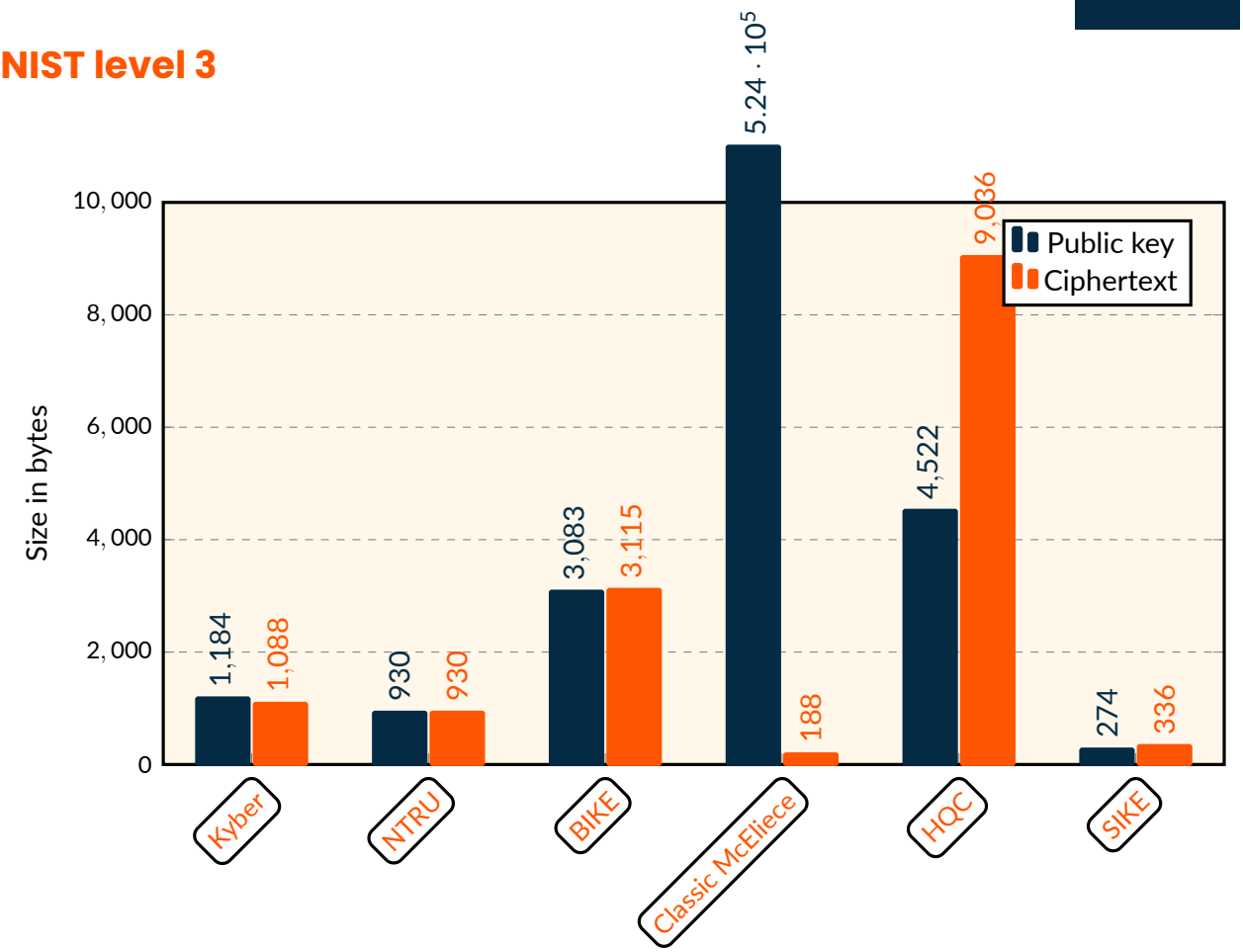
Communication Costs

The following section provides a detailed comparison of the communication costs of the 9 Round 3 KEMs for three security levels: NIST Level 1 (conjectured at least as secure as AES-128), NIST Level 3 (conjectured at least as secure as AES-192) and NIST Level 5 (conjectured at least as secure as AES-256). At the lowest security level (NIST Level 1), most schemes manage to keep their total communication cost below 2000 bytes. In that regard, the most efficient scheme is SIKE and the least efficient is Classic McEliece, which has very large public keys, although it manages to have the smallest ciphertexts across all schemes.

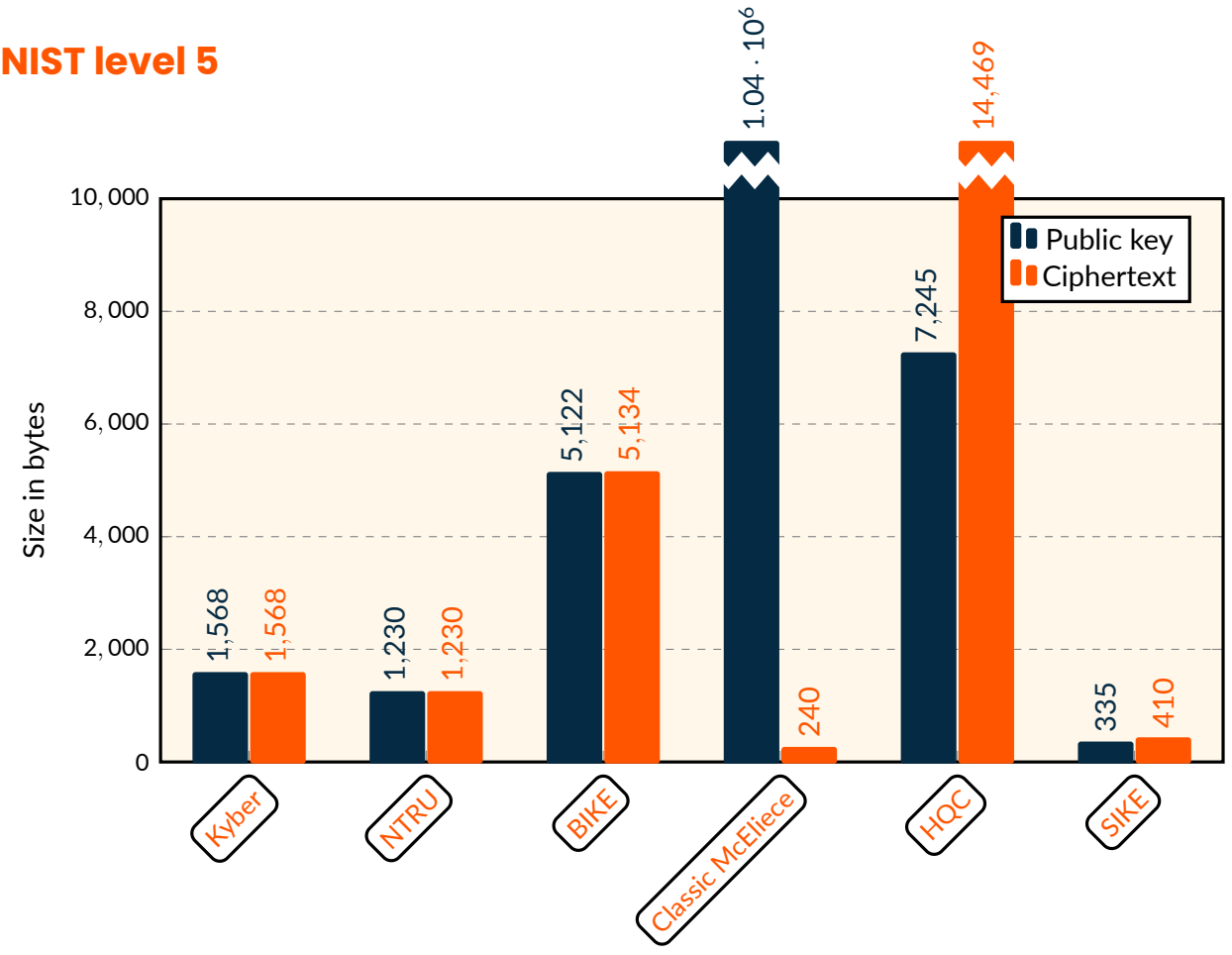
NIST level 1



NIST level 3



NIST level 5

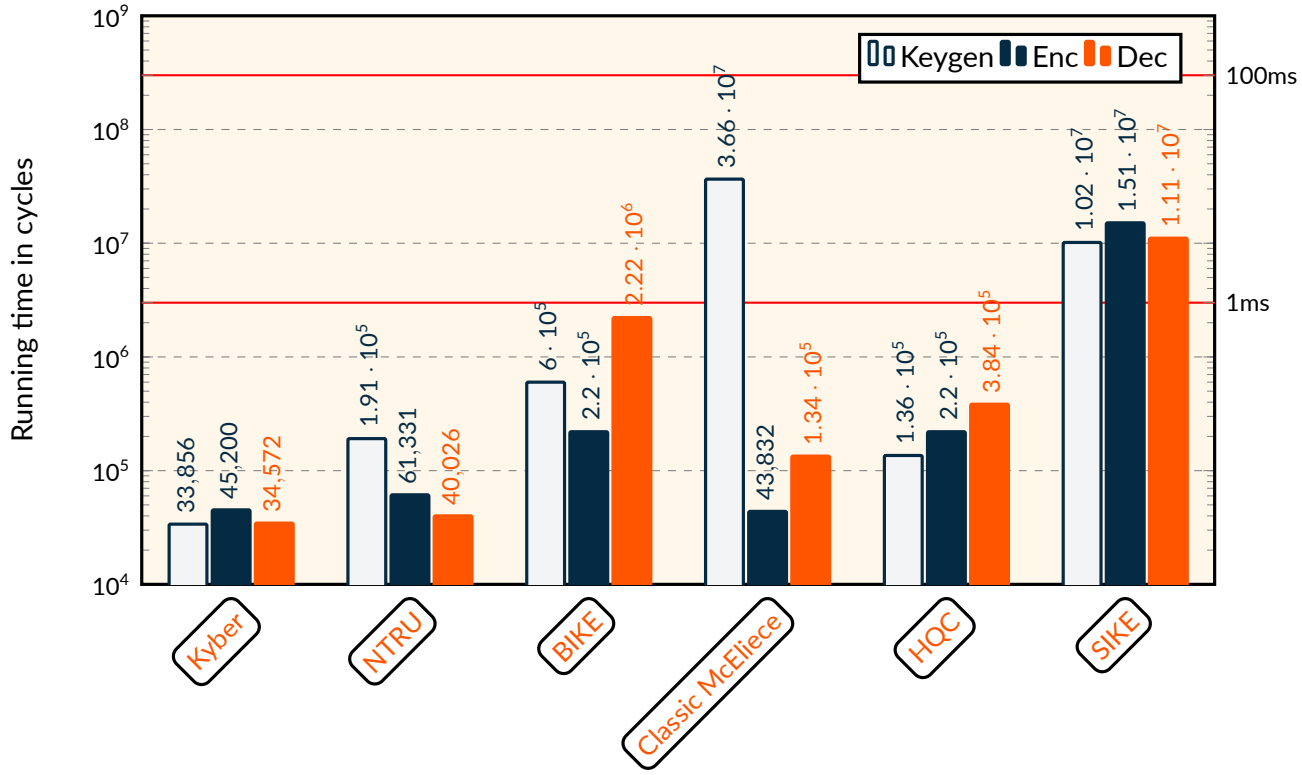


Computational Costs

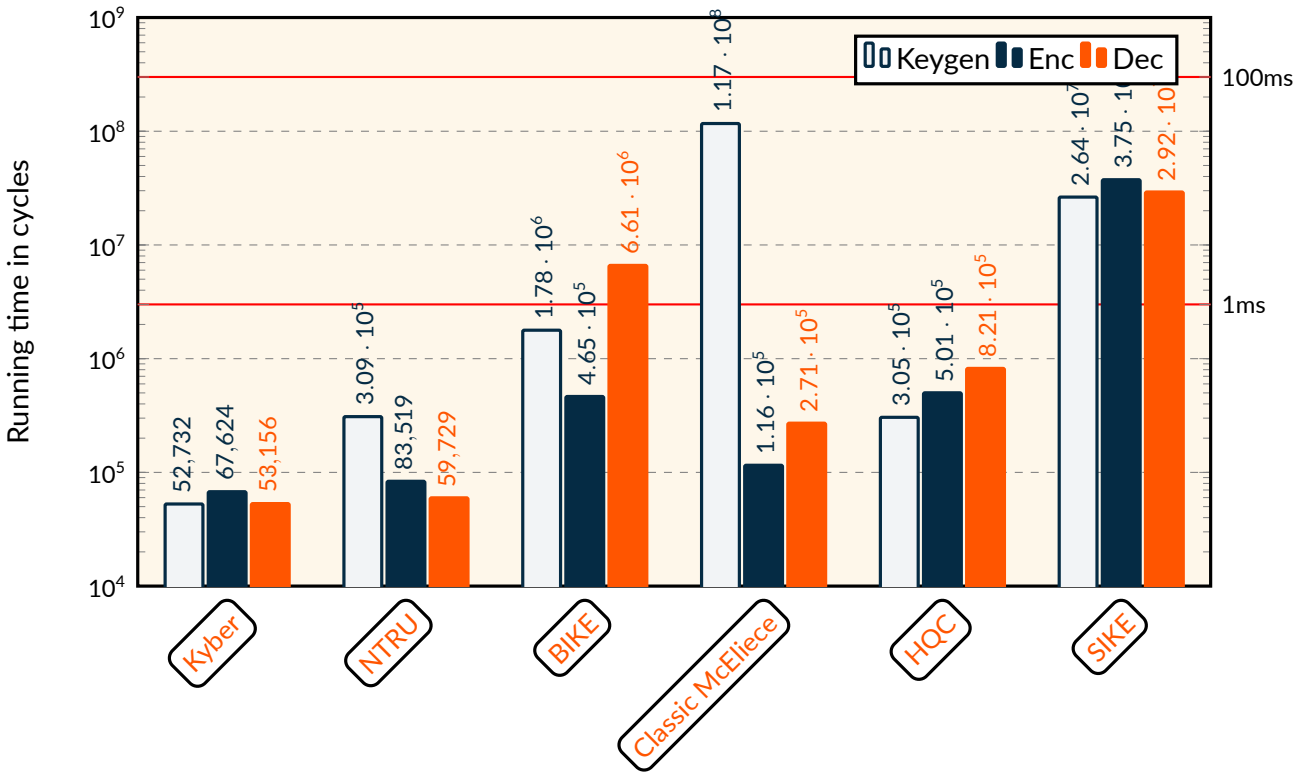
We now compare the running times in cycles of the 6 KEMs, for optimized implementations targeting x64 platforms. All numbers are extracted from the specification documents of the schemes (which might be inaccurate) and were obtained on different platforms. Therefore, they may not enable a completely fair comparison. To make these numbers less abstract, each graph also contains two horizontal red lines that correspond respectively to 1 and 100 milliseconds on a microprocessor with a clock frequency of 3GHz, which is typical for microprocessors in personal computers.

As with signatures, there can be a large disparity between candidates. At the highest security level, the fastest scheme overall (Kyber) is about 600 times faster than SIKE. On the other hand, SIKE is much cheaper in terms of bandwidth.

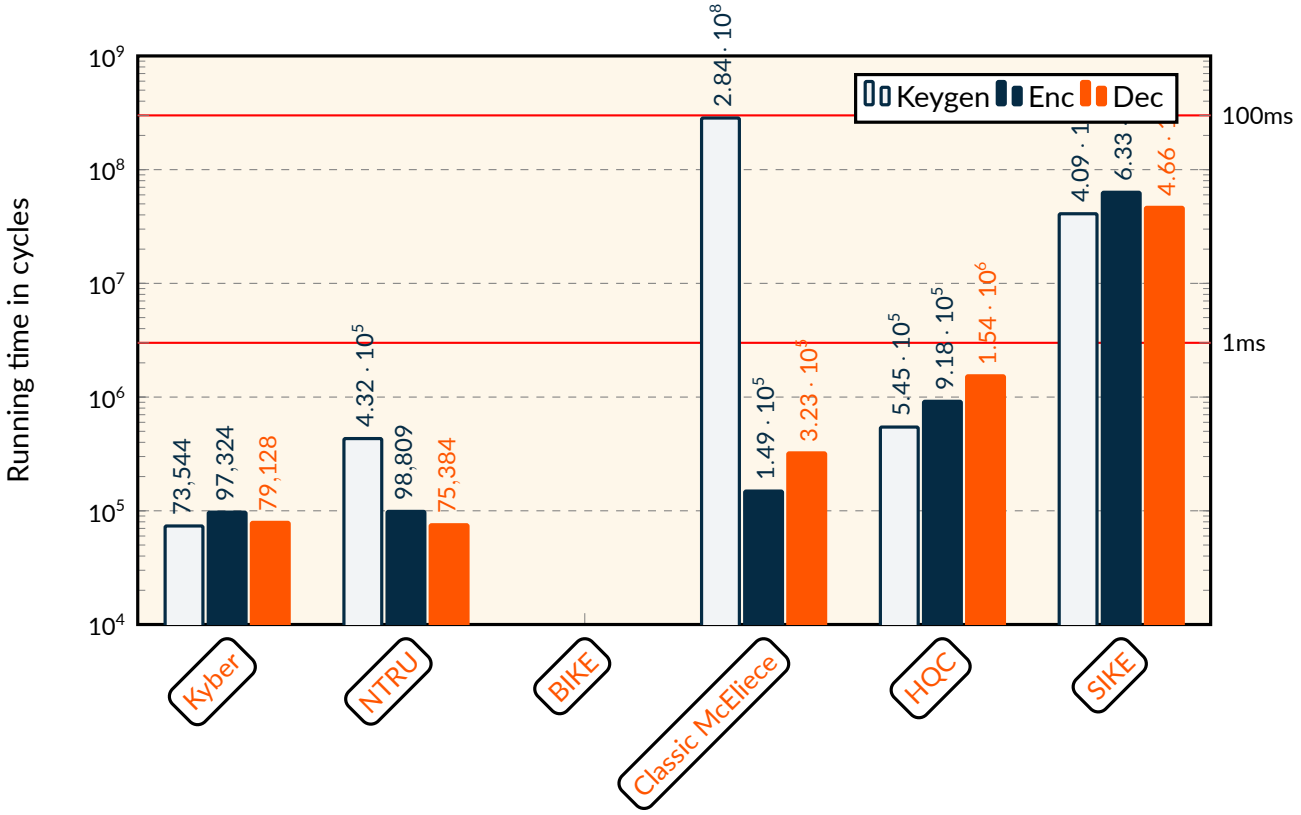
NIST level 1



NIST level 3



NIST level 5



Breakdown of Each Scheme

For each KEM, we now provide the following information:

- **The transform** is the generic conversion used to turn an IND-CPA scheme into an IND-CCA scheme. We recall that IND-CPA stands for *Indistinguishability under Chosen Plaintext Attack*, and IND-CCA stands for *Indistinguishability under Chosen Ciphertext Attack*. The former is simpler to achieve, but does not guarantee resistance against an attacker that can tamper with ciphertexts (for example in a man-in-the-middle attack). Therefore, IND-CPA schemes are usually converted to IND-CCA schemes using a *CCA transform*.
- **The family** can be either Error-correcting codes, Lattices or Isogenies.
- **The underlying hard problem** is specified.
- The **symmetric primitives** and the **type of randomness** used are specified. These can impact performance: in some schemes, the call to a symmetric primitive actually takes most of the running time. The type of randomness impacts how easy it is to protect a scheme against side-channel attacks, for example via the *masking* countermeasure.
- Links to **the specification**, **the website** (if any) and to **related works** are also provided.
- A **short summary** highlights the key facts about the scheme.
- Finally, a **performance table** is provided.

7 Kyber (Primary standard)

Type:	KEM
Paradigm:	Encryption
Family:	Lattices
Hard Problems:	Module-LWE
Sym. primitives:	SHA 3-256/512 and SHAKE-128/256
Randomness:	Binomial
Specification:	[SAB+20]
Website:	`https://pq-crystals.org/kyber/`
Related Works:	[LPR10, LP11, LS15, ADPS16b, ADPS16a, BDK+18]

NIST's overall assessment [NIS22]

"*The security of Kyber has been thoroughly analyzed and is based on a strong framework of results in lattice-based cryptography. Kyber has excellent performance overall in software, hardware and many hybrid settings.*"

Design

Kyber follows the Lindner-Peikert framework [LPR10, LP11], also used by Saber , FrodoKEM and NTRU Prime (NTRU LPRime). We give a simplified (CPA-secure) description below.

Key generation:

1. Sample a pseudo-random matrix $\mathbf{A}$.
2. Sample short matrices $\mathbf{S}, \mathbf{E}$.
3. Compute $\mathbf{B} = \mathbf{AS} + \mathbf{E}$.
4. The public key is $pk = (\mathbf{A}, \mathbf{B})$, and the private key is $sk = \mathbf{S}$.

Encryption:

1. Sample short matrices $\mathbf{R}, \mathbf{E}', \mathbf{E}''$.
2. Compute $\mathbf{U} = \mathbf{RA} + \mathbf{E}'$ and $\mathbf{V} = \mathbf{RB} + \mathbf{E}'' + Encode(msg)$.
3. The ciphertext is $ctxt = (\mathbf{U}, \mathbf{V})$.

Decryption:

1. $msg = Decode(\mathbf{V} - \mathbf{US})$.

Module lattices

Kyber uses *module lattices*: it manipulates matrices and vectors with entries in $\mathcal{R} = \mathbb{Z}_q[x]/(x^{256} + 1)$; the dimensions of these matrices and vectors are variable. This is meant to provide a trade-off between efficiency and conservatism, to make implementation simpler and to easily change security levels.

Hashing the public key

As is usual, CCA security is achieved by performing a variant of Fujisaki-Okamoto's transform. Kyber also hashes the public key as part of that process; it has been argued [BDK+18, SAB+20] that this provides protection against multi-target attacks and other useful properties.

Round 2 changes

Between the Round 1 and Round 2, Kyber has reduced the modulus q by a factor of about two, due to improvements in NTT techniques. The Round 1 version of Kyber [SAB+17] also included a technique for compressing public keys by dropping least significant bits, which was removed in Round 2.

NIST level	|SK| (bytes)	|PK| (bytes)	|sig| (bytes)	KG (cycles)	Sign (cycles)	Verify (cycles)
1	1632	800	768	33856	45200	34572
3	2400	1184	1088	52732	67624	53156
5	3168	1568	1568	73544	97324	79128

8 BIKE (Moves to Round 4)

Type: KEM
CCA Transform: FO$^{\not\perp}$ [HHK17]
Family: Error-correcting codes (QC-MDPC codes)
Hard Problems: Quasi-Cyclic Syndrome Decoding and Codeword Finding problems
Sym. primitives: AES, SHA
Randomness: Uniform, fixed weight, odd weight
Specification: [ABB+20]
Website: `https://bikesuite.org/`
Related Works: [MTSB12, BGG+17, HHK17]

	BIKE-1	BIKE-2	BIKE-3
SK	(h_0, h_1) with $\|h_0\| = \|h_1\| = w/2$		
PK	$(f_0, f_1) \leftarrow (gh_1, gh_0)$	$(f_0, f_1) \leftarrow (1, h_1 h_0^{-1})$	$(f_0, f_1) \leftarrow (h_1 + gh_0, g)$
Enc	$(c_0, c_1) \leftarrow (mf_0 + e_0, mf_1 + e_1)$	$c \leftarrow e_0 + e_1 f_1$	$(c_0, c_1) \leftarrow (e + e_1 f_0, e_0 + e_1 f_1)$
	$K \leftarrow \mathbf{K}(e_0, e_1)$		
Dec	$s \leftarrow c_0 h_0 + c_1 h_1$; $u \leftarrow 0$	$s \leftarrow c h_0$; $u \leftarrow 0$	$s \leftarrow c_0 + c_1 h_0$; $u \leftarrow t/2$
	$(e'_0, e'_1) \leftarrow \mathtt{Decode}(s, h_0, h_1, u)$		
	$K \leftarrow \mathbf{K}(e'_0, e'_1)$		

NIST's overall assessment [NIS22]

"BIKE has the most competitive performance among the non-lattice-based KEMs. [...] BIKE remains under consideration due to its overall performance and substantially different security assumption from the currently selected KEM."

Design and variants

BIKE is based on QC-MDPC (*Quasi-Cyclic Moderate Density Parity Check*) codes. This structure provides dramatic gains in compactness and speed. BIKE initially had three variants, presented in the above table extracted from the Round 1 presentation of BIKE. Only BIKE-2 was kept in the last iteration. BIKE-2 was the most compact of the three variants; it also has a much slower key generation, but this was partially addressed in [DGK20a].

The decoding algorithm

Decoding algorithms for code-based KEMs has been the topic of intensive research. Decryption failures have been shown [GJS16] to lead to practical attacks, hence the decryption failure rate (DFR) must be kept negligible. However, constant-time decoding algorithms with negligible DFR have been difficult to obtain. BIKE currently uses the Black-Gray-Flip decoder [DGK20b].

Hardware implementation

BIKE is one of the few Round 3 candidates to have proposed a hardware implementation (on Artix-7 FPGA), see: `https://github.com/Chair-for-Security-Engineering/BIKE`.

NIST level	|SK| (bytes)	|PK| (bytes)	|ctxt| (bytes)	KG (cycles)	Enc (cycles)	Dec (cycles)
1	281	1541	1573	600000	220000	2220000
3	419	3083	3115	1780000	465000	6610000
5	580	5122	5134	-	-	-

9 Classic McEliece (Moves to Round 4)

Type: KEM
CCA Transform: Dent [Den03], SXY [SXY18], see also [BP18a]
Family: Error-correcting codes (Goppa codes)
Hard Problems: Syndrome Decoding, Indistinguishability of Goppa codes from random codes
Sym. primitives: SHAKE
Randomness: Uniform, fixed weight
Specification: [ABC+20]
Website: `https://classic.mceliece.org`
Related Works: [McE78, Den03, Nie86, SXY18]

NIST's overall assessment [NIS22]

"*NIST is confident in the security of Classic McEliece and would be comfortable standardizing the submitted parameter sets [...]. However, it is unclear whether Classic McEliece represents the best option for enough applications to justify standardizing it at this time. [...] NIST would like feedback on specific use cases for which Classic McEliece would be a good solution.*"

Design

Despite its name, Classic McEliece is not exactly based on McEliece's scheme [McE78], but rather on a dual variant by Niederreiter [Nie86], which is equivalent security-wise. One of the selling points of Classic McEliece is its very conservative design: the original designs by [McE78, Nie86] have been extensively studied, and Classic McEliece makes no fundamental change to them.

Chosen-ciphertext security

Classic McEliece uses a different CCA transform than other schemes. This transform is inspired by Dent [Den03] and Saito-Xagawa-Yamakawa [SXY18]. See also [BP18a] for discussions on the QROM security of this transform.

Size constraints

Classic McEliece has very large public keys but very small ciphertexts. Although this may make it unsuitable in some contexts, applications for which ciphertext size is more important than key size may benefit from it. This is argued in [HNS+20], which uses Classic McEliece in a post-quantum version of the WireGuard protocol. See also [BL20] for a protocol built around these constraints.

Hardware implementation and attacks

Hardware implementations of the core mathematical elements of Classic McEliece have been provided in [WSN18], and the specification provides performance numbers on Artix-7 and Virted-7 FPGAs. Note that this is not a full implementation *per se* (it does not include, e.g., hashing).

The implementation of [WSN18] implements the Berlekamp-Massey decoder in constant-time to prevent timing attacks. However [LNPS20] showed that it is still vulnerable to an electromagnetic side-channel attack, and shows it is possible to recover a plaintext in a few hundred power traces.

NIST level	|SK| (bytes)	|PK| (bytes)	|ctxt| (bytes)	KG (cycles)	Enc (cycles)	Dec (cycles)
1	6492	261120	128	36627388	43832	134184
3	13608	524160	188	116914656	115540	270856
5	13932	1044992	240	284468140	149080	322988

10 HQC (Moves to Round 4)

Type: KEM
CCA Transform: Variant [HHK17] of FO
Family: Error-correcting codes
Hard Problems: Quasi-Cyclic Syndrome Decoding
Sym. primitives: AES, SHA
Randomness: Uniform, fixed weight
Specification: [AAB+20]
Website: `http://pqc-hqc.org`
Related Works: [Ale03, Gab05, ABD+16, DGZ17]

- Setup(1^λ): generates and outputs the global parameters param $= (n, k, \delta, w, w_{\mathbf{r}}, w_{\mathbf{e}})$.
- KeyGen(param): samples $\mathbf{h} \xleftarrow{\$} \mathcal{R}$, the generator matrix $\mathbf{G} \in \mathbb{F}_2^{k\times n}$ of $\mathcal{C}$, sk $= (\mathbf{x}, \mathbf{y}) \xleftarrow{\$} \mathcal{R}^2$ such that $\omega(\mathbf{x}) = \omega(\mathbf{y}) = w$, sets pk $= (\mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h}\cdot\mathbf{y})$, and returns (pk, sk).
- Encrypt(pk, $\mathbf{m}$): generates $\mathbf{e} \xleftarrow{\$} \mathcal{R}$, $\mathbf{r} = (\mathbf{r}_1, \mathbf{r}_2) \xleftarrow{\$} \mathcal{R}^2$ such that $\omega(\mathbf{e}) = w_{\mathbf{e}}$ and $\omega(\mathbf{r}_1) = \omega(\mathbf{r}_2) = w_{\mathbf{r}}$, sets $\mathbf{u} = \mathbf{r}_1 + \mathbf{h}\cdot\mathbf{r}_2$ and $\mathbf{v} = \mathbf{mG} + \mathbf{s}\cdot\mathbf{r}_2 + \mathbf{e}$, returns $\mathbf{c} = (\mathbf{u}, \mathbf{v})$.
- Decrypt(sk, $\mathbf{c}$): returns $\mathcal{C}$.Decode$(\mathbf{v} - \mathbf{u}\cdot\mathbf{y})$.

NIST's overall assessment [NIS22]

"The overall performance of HQC is acceptable, though not optimal. [...] HQC remains under consideration due to the rigorous security analysis and substantially different security assumption from the currently selected KEM."

Design

HQC stands for Hamming Quasi-Cyclic. Just like BIKE, HQC relies on quasi-cyclic codes. Its high-level design is presented above. Lattice practitioners will recognize a design similar to lattice-based schemes such as the future standard Kyber. While this analogy can be useful at a very high level, the mathematical objets used are different (codes vs lattices) and therefore HQC relies on completely different problems and algorithms.

Attacks against the BCH decoder

Implementation attacks were proposed against the BCH decoder used in earlier versions of HQC. In [WTBB+19], co-authors of HQC displayed a timing attack exploiting the BCH decoder running time, and proposed a constant-time variant as a countermeasure. [SRSWZ20] mounted a power side-channel against the BCH decoder. The last iteration of HQC has replaced the BCH decoder with a Reed-Muller Reed-Solomon decoder.

A decryption failure attack

A decryption failure attack against a Round 2 parameter set of HQC has been proposed in [GJ20]. This parameter set is not present in the last iteration of HQC.

NIST level	|SK| (bytes)	|PK| (bytes)	|ctxt| (bytes)	KG (cycles)	Enc (cycles)	Dec (cycles)
1	40	2249	4481	136000	220000	384000
3	40	4522	9036	305000	501000	821000
5	40	7245	14469	545000	918000	1538000

11 SIKE (Moves to Round 4)

Type: Key exchange and KEM
CCA Transform: Variant of [HHK17]
Family: Isogenies
Hard Problems: SIDH problem
Sym. primitives: SHAKE-256
Randomness: Uniform
Specification: [JAC+20]
Website: `https://sike.org/`
Related Works: [JD11, CJL+17, CLN16, JS19, CLN+20, MLRB20]

NIST's overall assessment [NIS22]

"SIKE is an unusual candidate, as it relies on a different hard problem than all of the other post-quantum cryptosystems being evaluated by NIST. In terms of performance, it has both advantages (small key sizes) and disadvantages (slow running times). SIKE seems promising but needs further study, as it is still a relatively new scheme."

History and design

SIKE it implements the isogeny-based SIDH key-exchange [JD11]. Unlike classical Diffie-Hellman, it is not fully interactive. It is to be noted that while SIKE is the KEM with the lowest communication cost, it is one of those with the higher computational costs.

Compressed variant

SIKE comes in two variants, a basic one, and a second one that uses point compression [CJL+17], which reduces the public key size by about 41%, but multiplies the overall running time by about a factor of two. Our performance figures are for the variant with point compression.

Implementations

SIKE has attracted several implementations for embedded devices, including over ARM processors [SLLH18, sJA19], Xilinx Artix-7, Virtex-7, and Kintex UltraScale+ FPGAs [KAK18, KAK+19, MLRB20] or even for the RISC architecture [KPHS18]. Although SIKE is slower than other candidates, recent works consistently report running times of a few dozens milliseconds over these platforms.

Key-recovery attack

In July 2022, Castryck and Decru published a devastating key-recovery attack on SIKE [CD22]. It breaks all parameter sets in less than 24 hours on a laptop. As of August 2022, it is unclear whether SIKE can be repaired.

Other cryptanalysis

Prior to [CD22], the current best attack against SIKE was via claw-finding. The best classical algorithm is due to van Oorschot and Wiener [vW99], and the best quantum one to Tani. Jaques and Schanck [JS19] recently showed that in reasonable computation models, the classical attack is better than the quantum one. See also [CLN+20] for a state-of-the-art analysis.

NIST level	|SK| (bytes)	|PK| (bytes)	|ctxt| (bytes)	KG (cycles)	Enc (cycles)	Dec (cycles)
1	350	197	236	10158000	15120000	11077000
3	491	274	336	26360000	37470000	29216000
5	602	335	410	40935000	63254000	46606000

12 NTRU (Back-up)

Type:	KEM (and Encryption)
CCA Transform:	$U^{\not\perp}_m$ [HHK17], Saito-Xagawa-Yamada [SXY18], SimpleKEM [BP18a]
Family:	Lattices
Hard Problems:	One-Wayness under Chosen Plaintext Attacks (OW-CPA) of the underlying DPKE
Sym. primitives:	SHAKE-256, SHA 3-256
Randomness:	Ternary polynomials (sometimes with bounded weight)
Specification:	[CDH+20]
Website:	`https://ntru.org/`
Related Works:	[HPS98, Den03, HPS+17, HRSS17, Sch18]

NIST's overall assessment [NIS22]

"One important feature of NTRU is that because it has been around for longer, its IP situation is more clearly understood. [..] As noted by the submitters, NTRU may not be the fastest or smallest among the lattice KEM finalists, and for most applications and use cases, the performance would not be a problem. Nonetheless, as NIST has selected Kyber for standardization, NTRU will therefore not be considered for standardization in the fourth round."

History

NTRU has a long story as it was first proposed 20 years ago [HPS98]. Since then, the scheme has known a few evolutions. It was the first scheme for which decryption failure attacks (a common caveat of many lattice-based KEMs) were highlighted [HNP+03], and a fix was proposed via the NAEP transform [HSSW03]. Over the years, updated parameters were proposed [HHHW09, HPS+17] to account for cryptanalytic advances.

Design

NTRU is based on a variant of the eponymous assumption. By tweaking the parameters of the original NTRU scheme [HPS98], it becomes easy to implement in constant time and eliminates decryption failures [HNP+03], "evaluate-at-1" attacks and invertibility checks. The CCA transform used by NTRU can be interpreted and proven in many ways, see e.g. [HHK17, SXY18, BP18a].

A merge of two schemes

NTRU is the merge of two Round 1 schemes: NTRU-HRSS-KEM [SHRS17] and NTRUEncrypt [ZCHW17]. NTRU-HRSS-KEM aimed at perfect correctness and used a CCA transform inspired by Dent [Den03]. On the other hand, NTRUEncrypt relied on the NAEP transform [HSSW03], and proposed parameters with decryption failures, parameters inspired by a construction by Stehlé and Steinfeld [SS13] and (optionally) the use of Gaussian distributions.

Experimental TLS deployments

Google [Lan18] and Cloudflare [Kwi19] have experimentally deployed NTRU-HRSS-KEM (as well as SIKE) on TLS as an effort to assess the feasibility of a post-quantum TLS. Conclusions can be found at [KV19]. Similar deployment efforts were conducted by Amazon [Hop19, Wei20] on BIKE and SIKE.

NIST level	|SK| (bytes)	|PK| (bytes)	|ctxt| (bytes)	KG (cycles)	Enc (cycles)	Dec (cycles)
1	935	699	699	191279	61331	40026
3	1234	930	930	309216	83519	59729
5	1590	1230	1230	431667	98809	75384

References

[AAB+20] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, Gilles Zémor, and Jurjen Bos. HQC. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[ABB+20] Nicolas Aragon, Paulo Barreto, Slim Bettaieb, Loic Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Phillipe Gaborit, Shay Gueron, Tim Guneysu, Carlos Aguilar Melchor, Rafael Misoczki, Edoardo Persichetti, Nicolas Sendrier, Jean-Pierre Tillich, Gilles Zémor, Valentin Vasseur, and Santosh Ghosh. BIKE. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[ABC+20] Martin R. Albrecht, Daniel J. Bernstein, Tung Chou, Carlos Cid, Jan Gilcher, Tanja Lange, Varun Maram, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Kenneth G. Paterson, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, Cen Jung Tjhai, Martin Tomlinson, and Wen Wang. Classic McEliece. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[ABD+16] Carlos Aguilar, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, and Gilles Zémor. Efficient encryption from random quasi-cyclic codes. Cryptology ePrint Archive, Report 2016/1194, 2016. https://eprint.iacr.org/2016/1194.

[ADPS16a] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. NewHope without reconciliation. Cryptology ePrint Archive, Report 2016/1157, 2016. https://eprint.iacr.org/2016/1157.

[ADPS16b] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange - A new hope. In Thorsten Holz and Stefan Savage, editors, *USENIX Security 2016*, pages 327–343. USENIX Association, August 2016.

[AE17] Jean-Phillippe Aumasson and Guillaume Endignoux. Gravity-SPHINCS. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

[AE18] Jean-Philippe Aumasson and Guillaume Endignoux. Improving stateless hash-based signatures. In Nigel P. Smart, editor, *CT-RSA 2018*, volume 10808 of *LNCS*, pages 219–242. Springer, Heidelberg, April 2018.

[Ale03] Michael Alekhnovich. More on average case vs approximation complexity. In *44th FOCS*, pages 298–307. IEEE Computer Society Press, October 2003.

[aut20] Various authors. Official comments (round 3) - sphincs+, 2020. hhttps://csrc.nist.gov/csrc/media/Projects/post-quantum-cryptography/documents/round-3/official-comments/Sphincs-Plus-round3-official-comment.pdf.

[BDE+18] Jonathan Bootle, Claire Delaplace, Thomas Espitau, Pierre-Alain Fouque, and Mehdi Tibouchi. LWE without modular reduction and improved side-channel attacks against BLISS. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part I*, volume 11272 of *LNCS*, pages 494–524. Springer, Heidelberg, December 2018.

[BDH11] Johannes A. Buchmann, Erik Dahmen, and Andreas Hülsing. XMSS - A practical forward secure signature scheme based on minimal security assumptions. In Bo-Yin Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, pages 117–129. Springer, Heidelberg, November / December 2011.

[BDK+18] Joppe W. 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 2018, London, United Kingdom, April 24-26, 2018*, pages 353–367. IEEE, 2018.

[BG14] Shi Bai and Steven D. Galbraith. An improved compression technique for signatures based on learning with errors. In Josh Benaloh, editor, *CT-RSA 2014*, volume 8366 of *LNCS*, pages 28–47. Springer, Heidelberg, February 2014.

[BGG+17] Paulo S. L. M. Barreto, Shay Gueron, Tim Güneysu, Rafael Misoczki, Edoardo Persichetti, Nicolas Sendrier, and Jean-Pierre Tillich. CAKE: Code-based algorithm for key encapsulation. In Máire O'Neill, editor, *16th IMA International Conference on Cryptography and Coding*, volume 10655 of *LNCS*, pages 207–226. Springer, Heidelberg, December 2017.

[BH19] Daniel J. Bernstein and Andreas Hülsing. Decisional second-preimage resistance: When does SPR imply PRE? In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part III*, volume 11923 of *LNCS*, pages 33–62. Springer, Heidelberg, December 2019.

[BHH+15] Daniel J. Bernstein, Daira Hopwood, Andreas Hülsing, Tanja Lange, Ruben Niederhagen, Louiza Papachristodoulou, Michael Schneider, Peter Schwabe, and Zooko Wilcox-O'Hearn. SPHINCS: Practical stateless hash-based signatures. In

Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 368–397. Springer, Heidelberg, April 2015.

[BHK+19] Daniel J. Bernstein, Andreas Hülsing, Stefan Kölbl, Ruben Niederhagen, Joost Rijneveld, and Peter Schwabe. The SPHINCS+ signature framework. In Lorenzo Cavallaro, Johannes Kinder, XiaoFeng Wang, and Jonathan Katz, editors, *ACM CCS 2019*, pages 2129–2146. ACM Press, November 2019.

[BHLY16] Leon Groot Bruinderink, Andreas Hülsing, Tanja Lange, and Yuval Yarom. Flush, gauss, and reload - A cache attack on the BLISS lattice-based signature scheme. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *CHES 2016*, volume 9813 of *LNCS*, pages 323–345. Springer, Heidelberg, August 2016.

[BL20] Daniel J. Bernstein and Tanja Lange. McTiny: Fast high-confidence post-quantum key erasure for tiny network servers. In Srdjan Capkun and Franziska Roesner, editors, *USENIX Security 2020*, pages 1731–1748. USENIX Association, August 2020.

[BMTR21] Nina Bindel, Sarah McCarthy, Geoffrey Twardokus, and Hanif Rahbari. Suitability of 3rd round signature candidates for vehicle-to-vehicle communication. In *Third PQC Standardization Conference*, 2021. https://csrc.nist.gov/events/2021/third-pqc-standardization-conference.

[BNG21] Luke Beckwith, Duc Tri Nguyen, and Kris Gaj. High-performance hardware implementation of crystals-dilithium. In *International Conference on Field-Programmable Technology, (IC)FPT 2021, Auckland, New Zealand, December 6-10, 2021*, pages 1–10. IEEE, 2021.

[BNG22] Luke Beckwith, Duc Tri Nguyen, and Kris Gaj. High-performance hardware implementation of lattice-based digital signatures. *IACR Cryptol. ePrint Arch.*, page 217, 2022.

[BP18a] Daniel J. Bernstein and Edoardo Persichetti. Towards KEM unification. Cryptology ePrint Archive, Report 2018/526, 2018. https://eprint.iacr.org/2018/526.

[BP18b] Leon Groot Bruinderink and Peter Pessl. Differential fault attacks on deterministic lattice signatures. *IACR TCHES*, 2018(3):21–43, 2018. https://tches.iacr.org/index.php/TCHES/article/view/7267.

[CD22] Wouter Castryck and Thomas Decru. An efficient key recovery attack on sidh (preliminary version). Cryptology ePrint Archive, Paper 2022/975, 2022. https://eprint.iacr.org/2022/975.

[CDH+20] Cong Chen, Oussama Danba, Jeffrey Hoffstein, Andreas Hulsing, Joost Rijneveld, John M. Schanck, Peter Schwabe, William Whyte, Zhenfei Zhang, Tsunekazu Saito, Takashi Yamakawa, and Keita Xagawa. NTRU. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[CJL+17] Craig Costello, David Jao, Patrick Longa, Michael Naehrig, Joost Renes, and David Urbanik. Efficient compression of SIDH public keys. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 679–706. Springer, Heidelberg, April / May 2017.

[CLN16] Craig Costello, Patrick Longa, and Michael Naehrig. Efficient algorithms for supersingular isogeny Diffie-Hellman. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 572–601. Springer, Heidelberg, August 2016.

[CLN+20] Craig Costello, Patrick Longa, Michael Naehrig, Joost Renes, and Fernando Virdia. Improved classical cryptanalysis of SIKE in practice. In Aggelos Kiayias, Markulf Kohlweiss, Petros Wallden, and Vassilis Zikas, editors, *PKC 2020, Part II*, volume 12111 of *LNCS*, pages 505–534. Springer, Heidelberg, May 2020.

[CMP18] Laurent Castelnovi, Ange Martinelli, and Thomas Prest. Grafting trees: A fault attack against the SPHINCS framework. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018*, pages 165–184. Springer, Heidelberg, 2018.

[CPS+20] Chitchanok Chuengsatiansup, Thomas Prest, Damien Stehlé, Alexandre Wallet, and Keita Xagawa. ModFalcon: Compact signatures based on module-NTRU lattices. In Hung-Min Sun, Shiuh-Pyng Shieh, Guofei Gu, and Giuseppe Ateniese, editors, *ASIACCS 20*, pages 853–866. ACM Press, October 2020.

[DDLL13] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal Gaussians. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 40–56. Springer, Heidelberg, August 2013.

[Den03] Alexander W. Dent. A designer's guide to KEMs. In Kenneth G. Paterson, editor, *9th IMA International Conference on Cryptography and Coding*, volume 2898 of *LNCS*, pages 133–151. Springer, Heidelberg, December 2003.

[DFMS19] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the fiat-shamir transformation in the quantum random-oracle model. Cryptology ePrint Archive, Report 2019/190, 2019. https://eprint.iacr.org/2019/190.

[DGK20a] Nir Drucker, Shay Gueron, and Dusan Kostic. Fast polynomial inversion for post quantum QC-MDPC cryptography. Cryptology ePrint Archive, Report 2020/298, 2020. https://eprint.iacr.org/2020/298.

[DGK20b] Nir Drucker, Shay Gueron, and Dusan Kostic. QC-MDPC decoders with several shades of gray. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, pages 35–50. Springer, Heidelberg, 2020.

[DGZ17] Jean-Christophe Deneuville, Philippe Gaborit, and Gilles Zémor. Ouroboros: A simple, secure and efficient key exchange protocol based on coding theory. In Tanja Lange and Tsuyoshi Takagi, editors, *Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017*, pages 18–34. Springer, Heidelberg, 2017.

[DKL+18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: A lattice-based digital signature scheme. *IACR TCHES*, 2018(1):238–268, 2018. https://tches.iacr.org/index.php/TCHES/article/view/839.

[DLP14] Léo Ducas, Vadim Lyubashevsky, and Thomas Prest. Efficient identity-based encryption over NTRU lattices. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 22–41. Springer, Heidelberg, December 2014.

[DP16] Léo Ducas and Thomas Prest. Fast fourier orthogonalization. In Sergei A. Abramov, Eugene V. Zima, and Xiao-Shan Gao, editors, *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC 2016, Waterloo, ON, Canada, July 19-22, 2016*, pages 191–198. ACM, 2016.

[EFG+22] Thomas Espitau, Pierre-Alain Fouque, François Gérard, Mélissa Rossi, Akira Takahashi, Mehdi Tibouchi, Alexandre Wallet, and Yang Yu. Mitaka: A simpler, parallelizable, maskable variant of falcon. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part III*, volume 13277 of *LNCS*, pages 222–253. Springer, Heidelberg, May / June 2022.

[EFGT17] Thomas Espitau, Pierre-Alain Fouque, Benoît Gérard, and Mehdi Tibouchi. Side-channel attacks on BLISS lattice-based signatures: Exploiting branch tracing against strongSwan and electromagnetic emanations in microcontrollers. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1857–1874. ACM Press, October / November 2017.

[Gab05] Philippe Gaborit. Shorter keys for code-based cryptography. 01 2005.

[GJ20] Qian Guo and Thomas Johansson. A new decryption failure attack against HQC. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part I*, volume 12491 of *LNCS*, pages 353–382. Springer, Heidelberg, December 2020.

[GJS16] Qian Guo, Thomas Johansson, and Paul Stankovski. A key recovery attack on MDPC with CCA security using decoding errors. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 789–815. Springer, Heidelberg, December 2016.

[GKPM18] Aymeric Genêt, Matthias J. Kannwischer, Hervé Pelletier, and Andrew McLauchlan. Practical fault injection attacks on SPHINCS. Cryptology ePrint Archive, Report 2018/674, 2018. https://eprint.iacr.org/2018/674.

[GKS21] Denisa O. C. Greconici, Matthias J. Kannwischer, and Daan Sprenkels. Compact dilithium implementations on cortex-m3 and cortex-m4. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2021(1):1–24, 2021.

[GLP12] Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In Emmanuel Prouff and Patrick Schaumont, editors, *CHES 2012*, volume 7428 of *LNCS*, pages 530–547. Springer, Heidelberg, September 2012.

[GMRR22] Morgane Guerreau, Ange Martinelli, Thomas Ricosset, and Mélissa Rossi. The hidden parallelepiped is back again: Power analysis attacks on falcon. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2022(3):141–164, 2022.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.

[HBD+20] Andreas Hulsing, Daniel J. Bernstein, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Panos Kampanakis, Stefan Kolbl, Tanja Lange, Martin M Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, Peter Schwabe, Jean-Philippe Aumasson, Bas Westerbaan, and Ward Beullens. SPHINCS+. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[HHHW09] Philip S. Hirschhorn, Jeffrey Hoffstein, Nick Howgrave-Graham, and William Whyte. Choosing NTRUEncrypt parameters in light of combined lattice reduction and MITM approaches. In Michel Abdalla, David Pointcheval, Pierre-Alain Fouque, and Damien Vergnaud, editors, *ACNS 09*, volume 5536 of *LNCS*, pages 437–455. Springer, Heidelberg, June 2009.

[HHK17] Dennis Hofheinz, Kathrin Hövelmanns, and Eike Kiltz. A modular analysis of the Fujisaki-Okamoto transformation. In Yael Kalai and Leonid Reyzin, editors, *TCC 2017, Part I*, volume 10677 of *LNCS*, pages 341–371. Springer, Heidelberg, November 2017.

[HHP+03] Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H. Silverman, and William Whyte. NTRUSIGN: Digital signatures using the NTRU lattice. In Marc Joye, editor, *CT-RSA 2003*, volume 2612 of *LNCS*, pages 122–140. Springer, Heidelberg, April 2003.

[HNP+03] Nick Howgrave-Graham, Phong Q. Nguyen, David Pointcheval, John Proos, Joseph H. Silverman, Ari Singer, and William Whyte. The impact of decryption failures on the security of NTRU encryption. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 226–246. Springer, Heidelberg, August 2003.

[HNS+20] Andreas Hülsing, Kai-Chun Ning, Peter Schwabe, Florian Weber, and Philip R. Zimmermann. Post-quantum WireGuard. Cryptology ePrint Archive, Report 2020/379, 2020. https://eprint.iacr.org/2020/379.

[Hop19] Andrew Hopkins. Post-quantum tls now supported in aws kms. AWS Security Blog, 2019. https://aws.amazon.com/fr/blogs/security/post-quantum-tls-now-supported-in-aws-kms/.

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In *ANTS*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer, 1998.

[HPS+17] Jeffrey Hoffstein, Jill Pipher, John M. Schanck, Joseph H. Silverman, William Whyte, and Zhenfei Zhang. Choosing parameters for NTRUEncrypt. In Helena Handschuh, editor, *CT-RSA 2017*, volume 10159 of *LNCS*, pages 3–18. Springer, Heidelberg, February 2017.

[HRS16] Andreas Hülsing, Joost Rijneveld, and Fang Song. Mitigating multi-target attacks in hash-based signatures. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 387–416. Springer, Heidelberg, March 2016.

[HRSS17] Andreas Hülsing, Joost Rijneveld, John M. Schanck, and Peter Schwabe. High-speed key encapsulation from NTRU. In Wieland Fischer and Naofumi Homma, editors, *CHES 2017*, volume 10529 of *LNCS*, pages 232–252. Springer, Heidelberg, September 2017.

[HSSW03] Nick Howgrave-Graham, Joseph H. Silverman, Ari Singer, and William Whyte. NAEP: Provable security in the presence of decryption failures. Cryptology ePrint Archive, Report 2003/172, 2003. https://eprint.iacr.org/2003/172.

[Hül13] Andreas Hülsing. W-OTS+ - shorter signatures for hash-based signature schemes. In Amr Youssef, Abderrahmane Nitaj, and Aboul Ella Hassanien, editors, *AFRICACRYPT 13*, volume 7918 of *LNCS*, pages 173–188. Springer, Heidelberg, June 2013.

[JAC+20] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Luca De Feo, Basil Hess, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, Vladimir Soukharev, David Urbanik, Geovandro Pereira, Koray Karabina, and Aaron Hutchinson. SIKE. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[JD11] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In Bo-Yin Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, pages 19–34. Springer, Heidelberg, November / December 2011.

[JS19] Samuel Jaques and John M. Schanck. Quantum cryptanalysis in the RAM model: Claw-finding attacks on SIKE. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 32–61. Springer, Heidelberg, August 2019.

[KA21] Emre Karabulut and Aydin Aysu. FALCON down: Breaking FALCON post-quantum signature scheme through side-channel attacks. In *58th ACM/IEEE Design Automation Conference, DAC 2021, San Francisco, CA, USA, December 5-9, 2021*, pages 691–696. IEEE, 2021.

[KAK18] B. Koziel, R. Azarderakhsh, and M. M. Kermani. A high-performance and scalable hardware architecture for isogeny-based cryptography. *IEEE Transactions on Computers*, 67(11):1594–1609, Nov 2018.

[KAK+19] Brian Koziel, A-Bon Ackie, Rami El Khatib, Reza Azarderakhsh, and Mehran Mozaffari-Kermani. Sike'd up: Fast and secure hardware architectures for supersingular isogeny key encapsulation. Cryptology ePrint Archive, Report 2019/711, 2019. https://eprint.iacr.org/2019/711.

[KGB+18] Matthias J. Kannwischer, Aymeric Genêt, Denis Butin, Juliane Krämer, and Johannes Buchmann. Differential power analysis of XMSS and SPHINCS. In Junfeng Fan and Benedikt Gierlichs, editors, *COSADE 2018*, volume 10815 of *LNCS*, pages 168–188. Springer, Heidelberg, April 2018.

[KLS18] Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of Fiat-Shamir signatures in the quantum random-oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 552–586. Springer, Heidelberg, April / May 2018.

[KPHS18] Philipp Koppermann, Eduard Pop, Johann Heyszl, and Georg Sigl. 18 seconds to key exchange: Limitations of supersingular isogeny diffie-hellman on embedded devices. Cryptology ePrint Archive, Report 2018/932, 2018. https://eprint.iacr.org/2018/932.

[KV19] Kris Kwiatkowski and Luke Valenta. Tthe tls post-quantum experiment. Cloudflare Blog, 2019. https://blog.cloudflare.com/the-tls-post-quantum-experiment/.

[Kwi19] Kris Kwiatkowski. Towards post-quantum cryptography in tls. Cloudflare Blog, 2019. https://blog.cloudflare.com/towards-post-quantum-cryptography-in-tls/.

[Lan18] Adam Langley. CECPQ2. Imperial Violet, 2018. https://www.imperialviolet.org/2018/12/12/cecpq2.html.

[LAZ19] Xingye Lu, Man Ho Au, and Zhenfei Zhang. Raptor: A practical lattice-based (linkable) ring signature. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 19*, volume 11464 of *LNCS*, pages 110–130. Springer, Heidelberg, June 2019.

[LDK+20] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[LNPS20] Norman Lahr, Ruben Niederhagen, Richard Petri, and Simona Samardjiska. Side channel information set decoding using iterative chunking - plaintext recovery from the "classic McEliece" hardware reference implementation. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part I*, volume 12491 of *LNCS*, pages 881–910. Springer, Heidelberg, December 2020.

[LP11] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *CT-RSA 2011*, volume 6558 of *LNCS*, pages 319–339. Springer, Heidelberg, February 2011.

[LP19] Gaëtan Leurent and Thomas Peyrin. From collisions to chosen-prefix collisions application to full SHA-1. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 527–555. Springer, Heidelberg, May 2019.

[LP20] Gaëtan Leurent and Thomas Peyrin. SHA-1 is a shambles: First chosen-prefix collision on SHA-1 and application to the PGP web of trust. In Srdjan Capkun and Franziska Roesner, editors, *USENIX Security 2020*, pages 1839–1856. USENIX Association, August 2020.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, Heidelberg, May / June 2010.

[LS15] Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptography*, 75(3):565–599, 2015.

[LSG21] Georg Land, Pascal Sasdrich, and Tim Güneysu. A hard crystal - implementing dilithium on reconfigurable hardware. In Vincent Grosso and Thomas Pöppelmann, editors, *Smart Card Research and Advanced Applications - 20th International Conference, CARDIS 2021, Lübeck, Germany, November 11-12, 2021, Revised Selected Papers*, volume 13173 of *Lecture Notes in Computer Science*, pages 210–230. Springer, 2021.

[Lyu09] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Heidelberg, December 2009.

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755. Springer, Heidelberg, April 2012.

[LZ19] Qipeng Liu and Mark Zhandry. Revisiting post-quantum fiat-shamir. Cryptology ePrint Archive, Report 2019/262, 2019. https://eprint.iacr.org/2019/262.

[McE78] Robert J. McEliece. A public-key cryptosystem based on algebraic coding theory. The deep space network progress report 42-44, Jet Propulsion Laboratory, California Institute of Technology, January/February 1978. https://ipnpr.jpl.nasa.gov/progress_report2/42-44/44N.PDF.

[MGTF19] Vincent Migliore, Benoît Gérard, Mehdi Tibouchi, and Pierre-Alain Fouque. Masking Dilithium - efficient implementation and side-channel evaluation. In Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung, editors, *ACNS 19*, volume 11464 of *LNCS*, pages 344–362. Springer, Heidelberg, June 2019.

[MLRB20] Pedro Maat C. Massolino, Patrick Longa, Joost Renes, and Lejla Batina. A compact and scalable hardware/software co-design of SIKE. *IACR TCHES*, 2020(2):245–271, 2020. https://tches.iacr.org/index.php/TCHES/article/view/8551.

[MTSB12] Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo S. L. M. Barreto. MDPC-McEliece: New McEliece variants from moderate density parity-check codes. Cryptology ePrint Archive, Report 2012/409, 2012. https://eprint.iacr.org/2012/409.

[Nie86] H. Niederreiter. Knapsack-type cryptosystems and algebraic coding theory. *Problems of Control and Information Theory*, 15(2):159–166, 1986.

[NIS16] NIST. Submission requirements and evaluation criteria for the post-quantum cryptography standardization process, 2016. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf.

[NIS17] NIST. Post-quantum cryptography - round 1 submissions, 2017. https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

[NIS19] NIST. NISTIR 8240 - status report on the first round of the nist post-quantum cryptography standardization process, 2019. https://doi.org/10.6028/NIST.IR.8240.

[NIS20] NIST. Nistir 8309 - status report on the second round of the nist post-quantum cryptography standardization process, 2020. https://doi.org/10.6028/NIST.IR.8309.

[NIS22] NIST. Nistir 8413 - status report on the third round of the nist post-quantum cryptography standardization process, 2022. https://doi.org/10.6028/NIST.IR.8413.

[NR06] Phong Q. Nguyen and Oded Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 271–288. Springer, Heidelberg, May / June 2006.

[OSHG19] Tobias Oder, Julian Speith, Kira Höltgen, and Tim Güneysu. Towards practical microcontroller implementation of the signature scheme Falcon. In Jintai Ding and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 10th International Conference, PQCrypto 2019*, pages 65–80. Springer, Heidelberg, 2019.

[PBY17] Peter Pessl, Leon Groot Bruinderink, and Yuval Yarom. To BLISS-B or not to be: Attacking strongSwan's implementation of post-quantum signatures. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1843–1855. ACM Press, October / November 2017.

[PFH+20] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[Por19] Thomas Pornin. New efficient, constant-time implementations of Falcon. Cryptology ePrint Archive, Report 2019/893, 2019. https://eprint.iacr.org/2019/893.

[PP19] Thomas Pornin and Thomas Prest. More efficient algorithms for the NTRU key generation using the field norm. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 504–533. Springer, Heidelberg, April 2019.

[RMJ+21] Sara Ricci, Lukas Malina, Petr Jedlicka, David Smékal, Jan Hajny, Peter Cíbik, Petr Dzurenda, and Patrik Dobias. Implementing crystals-dilithium signature scheme on fpgas. In Delphine Reinhardt and Tilo Müller, editors, *ARES 2021: The 16th International Conference on Availability, Reliability and Security, Vienna, Austria, August 17-20, 2021*, pages 1:1–1:11. ACM, 2021.

[SAB+17] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, and Damien Stehlé. CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

[SAB+20] Peter Schwabe, Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, and Damien Stehlé. CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2020. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[SBK+17] Marc Stevens, Elie Bursztein, Pierre Karpman, Ange Albertini, and Yarik Markov. The first collision for full SHA-1. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 570–596. Springer, Heidelberg, August 2017.

[Sch18] John M. Schanck. A comparison of NTRU variants. Cryptology ePrint Archive, Report 2018/1174, 2018. https://eprint.iacr.org/2018/1174.

[SHRS17] John M. Schanck, Andreas Hulsing, Joost Rijneveld, and Peter Schwabe. NTRU-HRSS-KEM. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

[sJA19] Hwajeong soe, Amir Jalali, and Reza Azarderakhsh. Sike round 2 speed record on arm cortex-m4. Cryptology ePrint Archive, Report 2019/535, 2019. https://eprint.iacr.org/2019/535.

[SKD20] Dimitrios Sikeridis, Panos Kampanakis, and Michael Devetsikiotis. Post-quantum authentication in TLS 1.3: A performance study. In *NDSS 2020*. The Internet Society, February 2020.

[SLLH18] Hwajeong Seo, Zhe Liu, Patrick Longa, and Zhi Hu. SIDH on ARM: Faster modular multiplications for faster post-quantum supersingular isogeny key exchange. *IACR TCHES*, 2018(3):1–20, 2018. https://tches.iacr.org/index.php/TCHES/article/view/7266.

[SRSWZ20] Thomas Schamberger, Julian Renner, Georg Sigl, and Antonia Wachter-Zeh. A power side-channel attack on the CCA2-secure HQC KEM. Cryptology ePrint Archive, Report 2020/910, 2020. https://eprint.iacr.org/2020/910.

[SS13] Damien Stehlé and Ron Steinfeld. Making NTRUEncrypt and NTRUSign as secure as standard worst-case problems over ideal lattices. Cryptology ePrint Archive, Report 2013/004, 2013. https://eprint.iacr.org/2013/004.

[SXY18] Tsunekazu Saito, Keita Xagawa, and Takashi Yamakawa. Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 520–551. Springer, Heidelberg, April / May 2018.

[vW99] Paul C. van Oorschot and Michael J. Wiener. Parallel collision search with cryptanalytic applications. *Journal of Cryptology*, 12(1):1–28, January 1999.

[Wei20] Alex Weibel. Round 2 hybrid post-quantum tls benchmarks. AWS Security Blog, 2020. https://aws.amazon.com/fr/blogs/security/round-2-hybrid-post-quantum-tls-benchmarks/.

[WSN18] Wen Wang, Jakub Szefer, and Ruben Niederhagen. FPGA-based niederreiter cryptosystem using binary goppa codes. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography - 9th International Conference, PQCrypto 2018*, pages 77–98. Springer, Heidelberg, 2018.

[WTBB+19] Guillaume Wafo-Tapa, Slim Bettaieb, Loic Bidoux, Philippe Gaborit, and Etienne Marcatel. A practicable timing attack against HQC and its countermeasure. Cryptology ePrint Archive, Report 2019/909, 2019. https://eprint.iacr.org/2019/909.

[ZCHW17] Zhenfei Zhang, Cong Chen, Jeffrey Hoffstein, and William Whyte. NTRUEncrypt. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.