Think openly, build securely

White paper:

Understanding the Upcoming NIST Post-Quantum Cryptographic Standards

PQShield

February 10, 2021

The National Institute of Standards and Technology (or NIST) is a US Government agency responsible for standardizing several cryptographic primitives that are now ubiquitous: DES, AES, HMAC, SHA-{1,2,3}, (EC)DSA and much more. NIST is currently in the final phase of the process of standardizing post-quantum cryptography [NIS17b, NIS19a, NIS20b] and the winners are expected to be announced early 2022.

This document is a presentation of the 15 schemes (7 signature schemes and 8 key-establishment schemes) that are still under consideration by NIST to be the upcoming post-quantum cryptographic standards. As a follow-up to our overview of post-quantum cryptography [PQS20], it provides a more practical view of post-quantum schemes, some of which will become standards for decades to come. We present each of the 15 schemes in detail, and also provide extensive comparisons between them: bandwidth cost, computational cost, hardness assumptions and so on.

Contents

1 The NIST Standardization Process 4

2 Signature Schemes 5

3 Dilithium 11

4 Falcon 12

5 Rainbow 13

6 GeMSS 14

7 Picnic 15

8 SPHINCS+ 16

9 Key-Encapsulation Mechanisms 17

10 Classic McEliece 23

11 Kyber 24

12 NTRU 25

13 Saber 26

14 BIKE 27

15 HQC 28

16 FrodoKEM 29

17 NTRU Prime 30

18 SIKE 31

1 The NIST Standardization Process

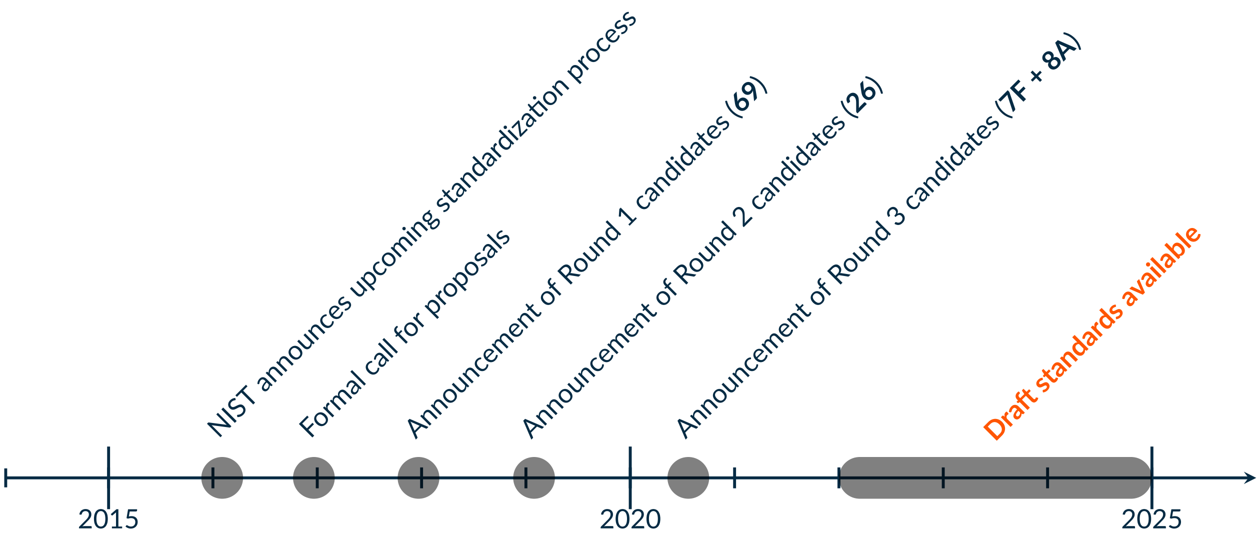
A question that may come to mind regarding standardization of post-quantum cryptography is:

Why 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, early standardization 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 February 2016 [NIS16], with the goal to:

- Standardize post-quantum signature schemes;
- Standardize post-quantum key-establishment schemes.



By the initial deadline of November 2017, 82 submissions were made. Out of these, 69 were considered "complete and proper" as per NIST's submission requirements and minimal acceptance criteria and were selected as Round 1 candidates (49 for key-establishment, 20 for signatures). In January 2019, 26 schemes were selected as Round 2 candidates (17 for key-establishment, 9 for signatures). In July 2020, 15 schemes were selected as Round 3 candidates. These are separated as *finalists* and *alternates*. As per NIST [NIS20b], "*finalists will be considered for standardization at the end of the third round*", while alternates "*are still being considered for standardization, although this is unlikely to occur at the end of the third round*".

Finally, NIST intends to make draft standards available between 2022 and 2024. Whatever NIST's choice, the draft standard(s) will be among the schemes presented in this document.

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

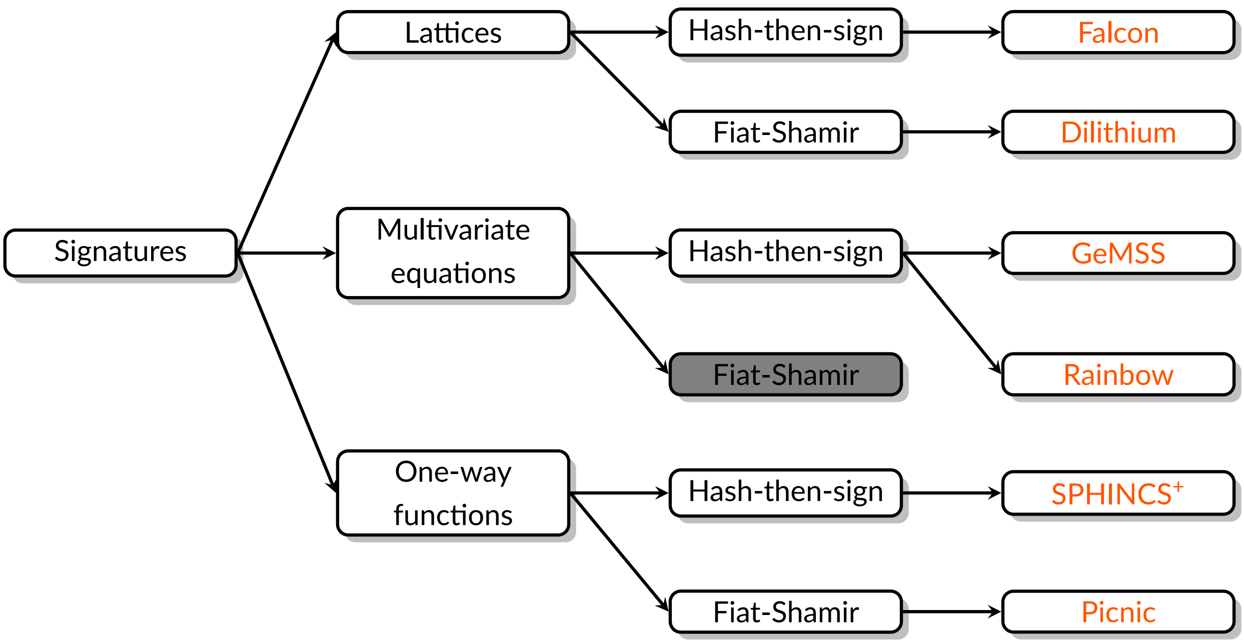
[2] Nature: *Quantum gold rush: the private funding pouring into quantum start-ups*

2 Signature Schemes

We first discuss signature schemes under consideration for standardization. In Round 1 of this process (December 2017), 20 digital signature schemes were accepted [NIS17b]. After a preliminary analysis by the cryptographic community, NIST selected 9 of these 20 schemes for Round 2 of the standardization process [NIS19a]. Out of these 9 schemes, 6 were selected for Round 3 of the standardization process [NIS20b]: 3 *finalists* and 3 *alternates*.

- Dilithium (finalist);
- Falcon (finalist);
- Rainbow (finalist);
- GeMSS (alternate);
- SPHINCS+ (alternate);
- Picnic (alternate).

These schemes are based on three families of hardness assumptions: lattices, multivariate equations, and one-way 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. An overview of the signature schemes can be found in the figure below.



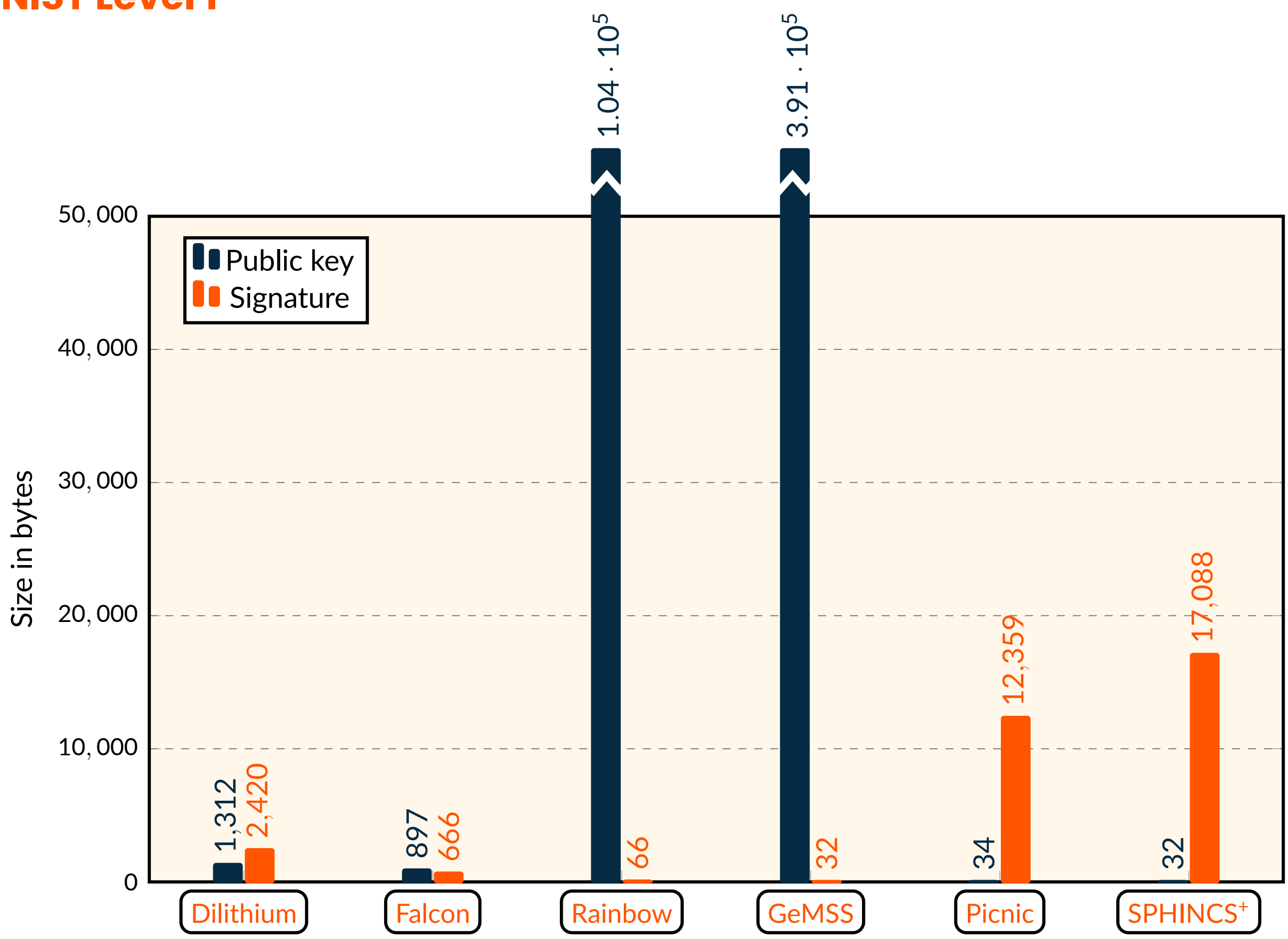
In addition, we provide the following data:

- A comparative performance study of the 6 signature schemes (pages 6 to 9);
- For each scheme, one page summarizing its main properties (pages 11 to 16).

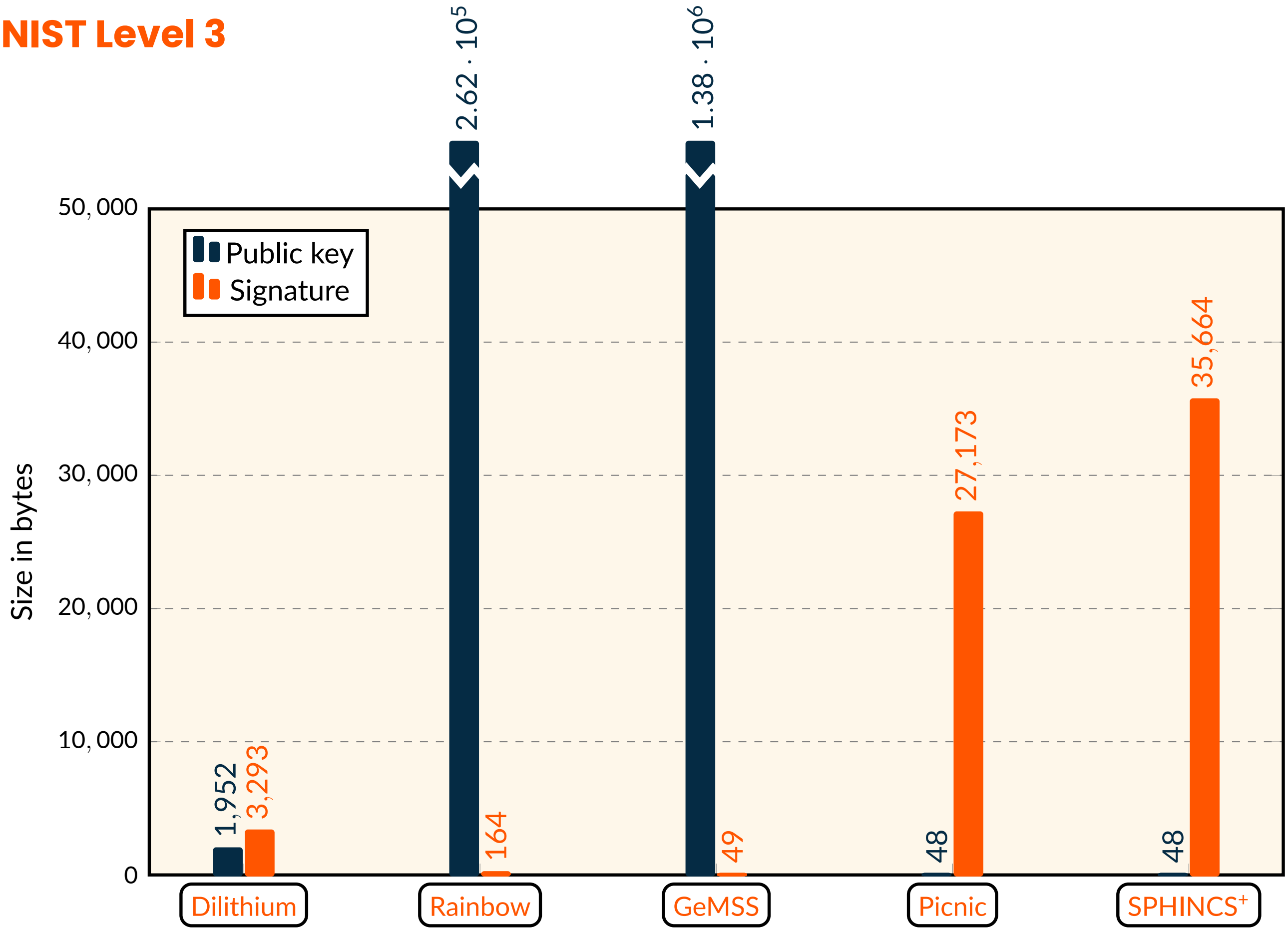
Communication Costs

We now provide a detailed comparison of the communication costs of the 6 signature schemes for three security levels: NIST Level 1, 3 and 5 (conjectured at least as secure as AES-128, AES-192 and AES-256, respectively). Interestingly, there can be huge differences between schemes: two schemes have small public keys and small signatures (lattice-based schemes Dilithium and Falcon), two provide extremely small public keys but large signatures (Picnic and SPHINCS^{+}) and two have enormous public keys but very small signatures (multivariate schemes GeMSS and Rainbow).

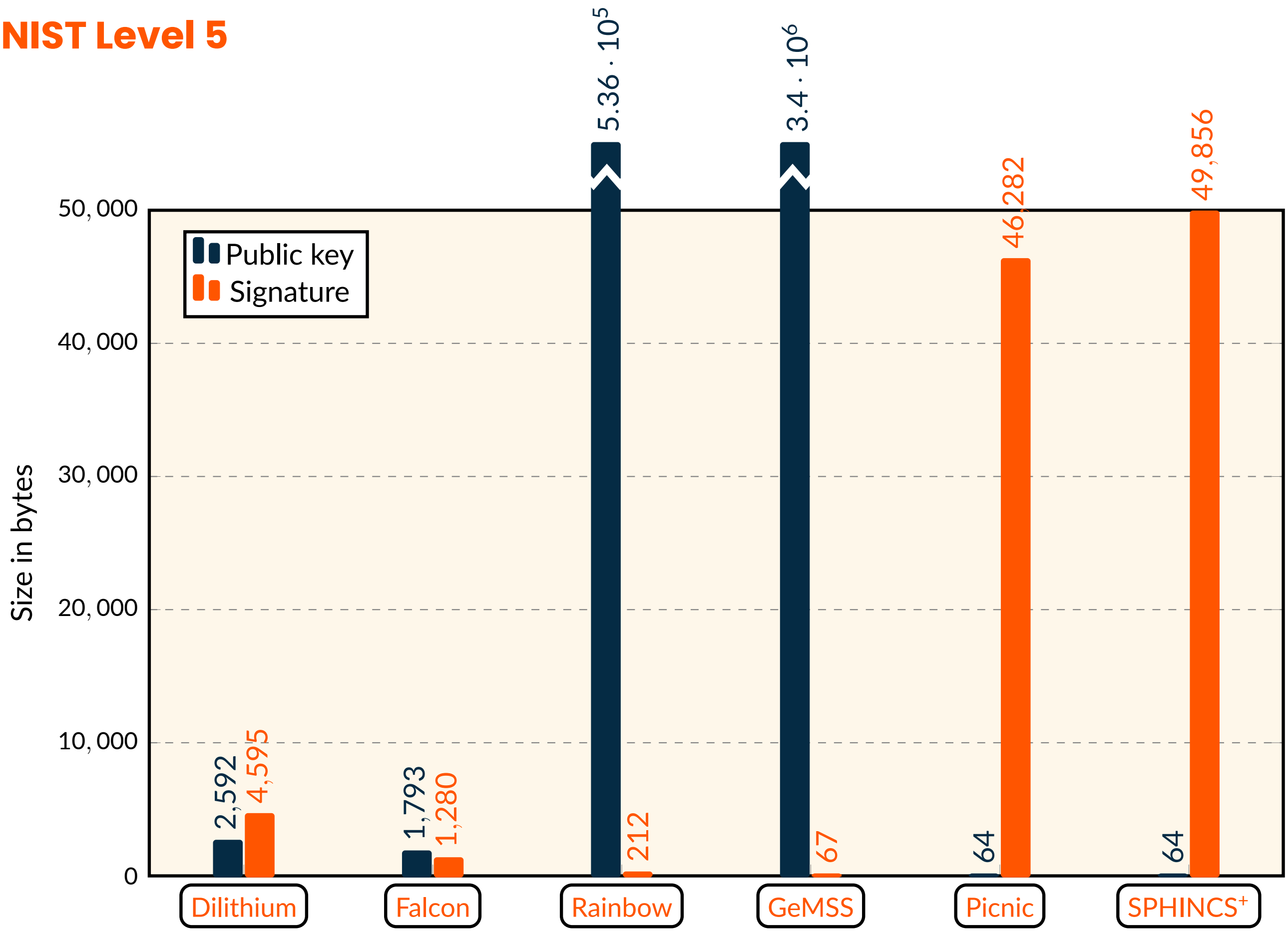
NIST Level 1



NIST Level 3



NIST Level 5

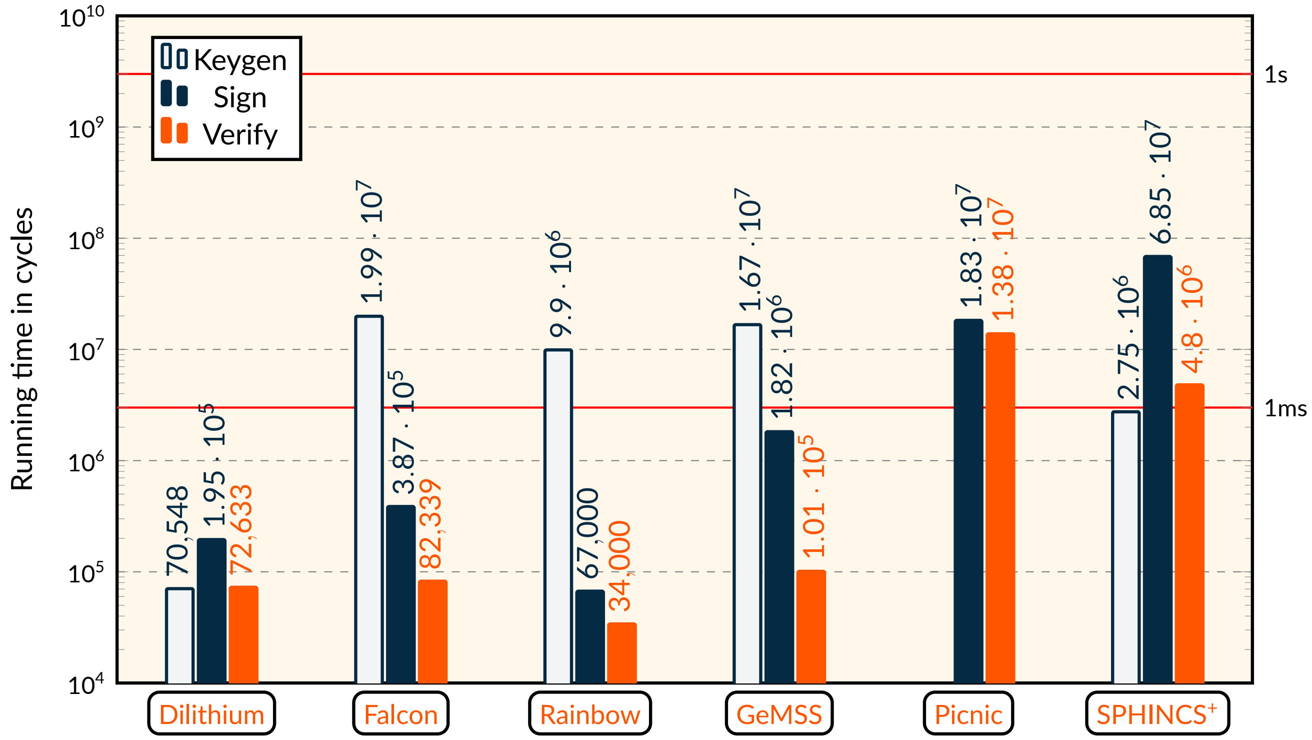


Computational Costs

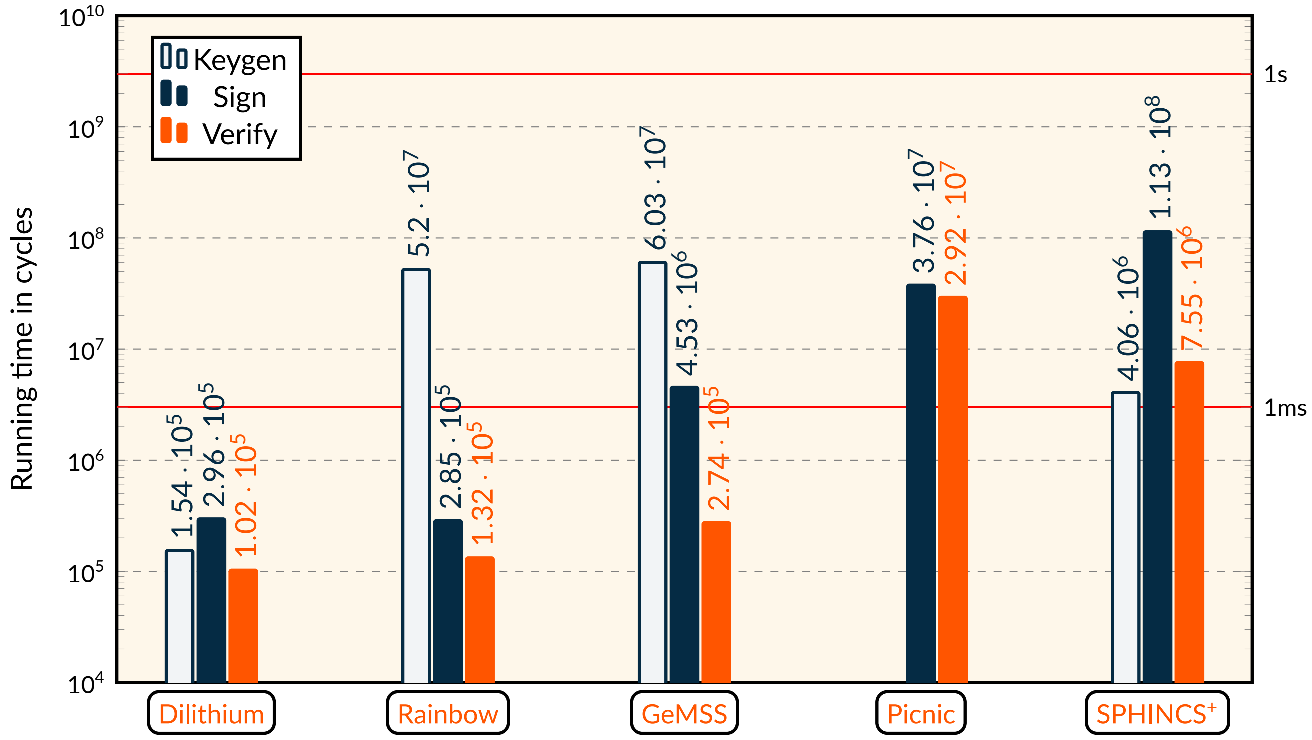
We now compare the running times in cycles of the 6 signature schemes, for optimized implementations targeting x64 platforms. All numbers are extracted from the specification documents of the schemes (which might be inaccurate) and were obtained on different platforms. Therefore, they may not enable 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.

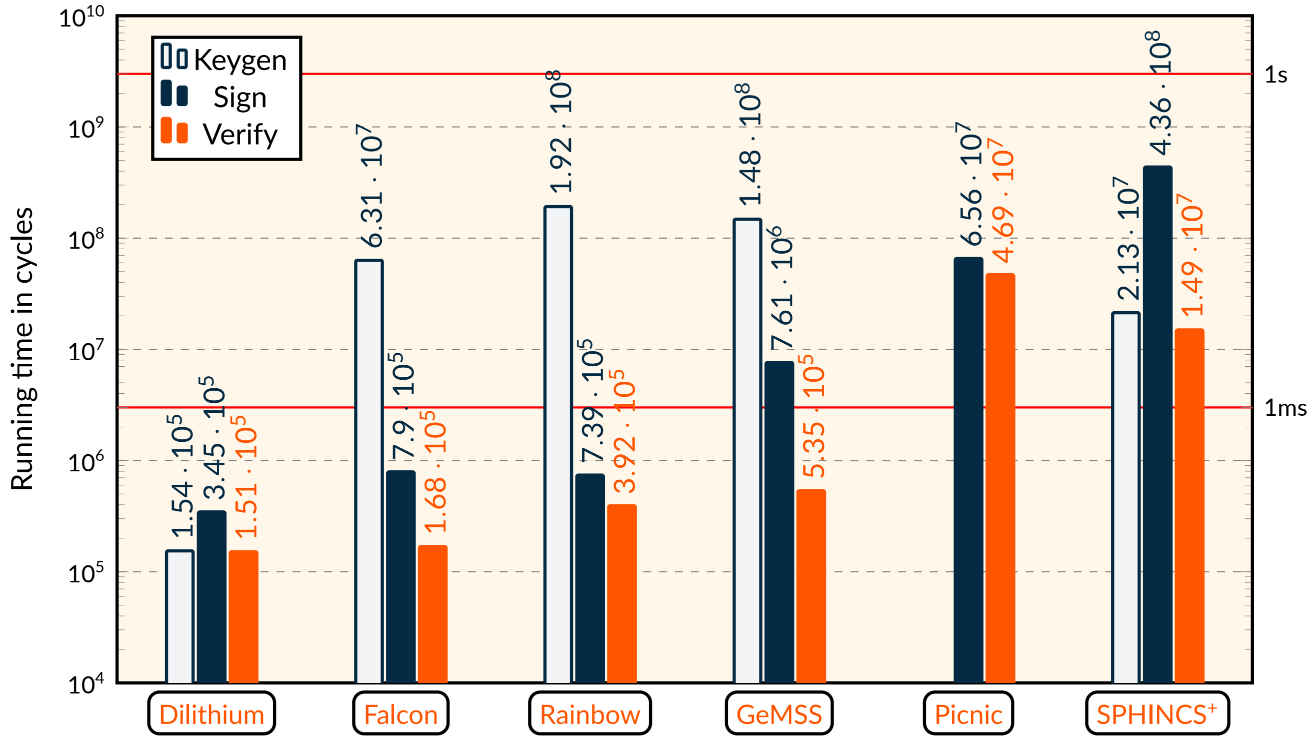
NIST Level 1



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, Multivariate equations or One-way functions.
- **The underlying hard problem(s)** is specified.
- The **symmetric primitives** and the **type of randomness** used are specified. While these are not too important theoretically, they can have a huge impact on performance. For example, schemes such as Picnic and SPHINCS^{+} are very dependent on the symmetric primitive used, 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 (finalist)

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]

Paradigm

Dilithium is based on the Fiat-Shamir with Aborts paradigm, introduced in [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 only one ring element instead of two. It also borrows elements of design from BLISS [DDLL13].

Hard Problems

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] (this problem might not be necessary after all, as discussed in the 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] seem to imply that the security proof of Dilithium can be made stronger, and that the hypothesis SelfTargetMSIS may not be necessary after all.

Design Rationale and Physical Attacks

The design of Dilithium has been heavily influenced by the numerous side-channel attacks to which its precedessor, BLISS, has been subjected [BHLY16, PBY17, EFGT17, BDE+18]. To guard against these attacks, Dilithium discards BLISS's use of Gaussian distributions, and relies on uniform distributions instead. A masked implementation of Dilithium has been proposed in [MGTF19].

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 (finalist)

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]

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 $\mathbb{Z}[x]/(x^n + 1)$.

Modes of Operation

The specification of Falcon highlights a few possible modes of operation; in addition to the classical mode, the key-recovery mode doubles the size of the signature but shrinks the key size to 64 bytes. The message-recovery mode recovers part of the message (similarly to RSA's key-recovery mode). Finally, Falcon can be converted to an identity-based encryption scheme *à la* [DLP14].

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.

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 Rainbow (finalist)

Type: Signature
Paradigm: Hash-then-sign
Family: Multivariate equations
Hard Problems: MQ (*Multivariate Quadratic*), UOV (*Oil and Vinegar*), MinRank
Sym. primitives: SHA (2), AES
Randomness: Uniform
Specification: [DCP+20]
Website: `https://www.pqcrainbow.org/`
Related Works: [DS05, PBB10, KPG99, SSH11, Pet20, Beu20]

Paradigm

Rainbow is based on the Hash-then-sign paradigm. It implements an upgraded version of UOV [KPG99] in order to propose more efficient parameters. For instance, for a specific choice of parameters one directly obtains the original UOV scheme.

Hard Problems

Rainbow relies on the MQ (*Multivariate Quadratic*) and UOV (*Oil and Vinegar*) problems. Note that this is heuristic, and there is no known security reduction of Rainbow to precisely defined instances of these problems.

Design Rationale

Rainbow is based on a quadratic central map and its inversion function. This inversion function is based on a parameter that can be seen as the depth or layer of inversion. Depending on the layer value, one can achieve different efficiency results for the underlying signature schemes: the more layers, the larger the key but the more efficient the implementation. It is worth noting that for a layer equal to one, the underlying signature scheme is the original UOV signature.

Variants

There are a few variants of Rainbow: standard Rainbow, CZ-Rainbow and compressed Rainbow. CZ-Rainbow uses a trick by [Pet20] to reduce the public key size, whereas the latter allows better memory usage with a compressed private key form but a slower signature generation process. Overall, Rainbow's main advantage is the shortness of the signatures. The performance numbers provided here are those of standard Rainbow.

Recent Attacks

Two papers [NIW+20, PS20] recently revisited the complexity of the *Rainbow Band Separation* attack. A new work by Beullens [Beu20] has shown that Rainbow failed to take into account another attack, and has put a significant dent in the security of currently proposed parameters.

NIST level	|SK| (bytes)	|PK| (bytes)	|sig| (bytes)	KG (cycles)	Sign (cycles)	Verify (cycles)
1	-	103628	66	9900000	67000	34000
3	-	261602	164	52000000	285000	132000
5	-	536166	212	192000000	739000	392000

6 GeMSS (alternate)

Type: Signature
Paradigm: Hash-then-sign
Family: Multivariate equations
Hard Problems: MinRank, HFEv-
Sym. primitives: SHA, AES
Randomness: Uniform, random invertible matrices
Specification: [CFM+20]
Website: `https://www-polsys.lip6.fr/Links/NIST/GeMSS.html`
Related Works: [PCG01, DY13, Pat96, KPG99]

Paradigm

GeMSS is based on the Hash-then-sign paradigm. It implements a direct lineage from QUARTZ [PCG01] and takes some design rationale from the Gui multivariate signature scheme [DY13]. Both schemes descend from the Hidden Field Equations cryptosystem.

Hard Problem

GeMSS relies on a variant of the Hidden Field Equations problem (HFE, [Pat96]). This variant, called HFEv-, was introduced in [PCG01] and adds two new parameters to HFE (*vinegar* and *minus* to HFE, hence the "v-").

Design and Variants

The design of GeMSS is heavily influenced by QUARTZ [PCG01], proposed in the early 2000's. It allows fast verification and short signature sizes at the cost of large public keys. The specification of GeMSS proposes 6 variants (GeMSS and Blue/Red/Cyan/White/Magenta-GeMSS), and each comes in 3 security levels, hence there are 18 parameter sets for GeMSS. The performance numbers we give are for MagentaGeMSS.

New Attack

A very recent work [TPD20] has shown that the *vinegar* and *minus* modifications only marginally increase the security of the HFE problem. As a consequence, they show that GeMSS, RedGeMSS and BlueGeMSS fall short of their claimed security levels.

NIST level	|SK| (bytes)	|PK| (bytes)	|sig| (bytes)	KG (cycles)	Sign (cycles)	Verify (cycles)
1	16	390615	32	16700000	1820000	101000
3	24	1380383	49	60300000	4530000	274000
5	32	3401441	67	148000000	7610000	535000

7 Picnic (alternate)

Type: Signature
Paradigm: Fiat-Shamir
Family: One-way functions
Hard Problems: Invertibility of a one-way function
Sym. primitives: LowMC (underlying one-way function), SHAKE
Randomness: Uniform
Specification: [ZCD+20]
Website: `https://microsoft.github.io/Picnic/`
Related Works: [ARS+15, CDG+17, KKW18, DN19, DKP+19]

High-Level Design

Picnic stands out among signature schemes because of its unique design, as it relies on multiparty computation (or MPC), a paradigm in which N parties collaborate to compute the output of a function F. The public key is $pk = F(sk)$, where F is a publicly-known one-way function. The signing procedure simulates a MPC protocol (this technique is called *MPC-in-the-head*, and is similar to what Fiat-Shamir does for sigma protocols), and the signature is a transcript of this simulation.

Picnic

Present in all three rounds, this version simulates a MPC protocol with 3 parties. It relies on the proof system ZKB++ [CDG+17], and its communication cost as well as its running times are comparable to those obtained by hash-based signatures.

Picnic2

Only present in Round 2 [ZCD+19], Picnic2 simulates a MPC protocol with 64 parties. It uses new techniques introduced by [KKW18], which divides the signature size by 3 but increases signing and verification times by an order of magnitude compared to Picnic.

Picnic3

Compared to Picnic2, this variant reduces the number of parties from 64 to 16, and makes some tweaks to the MPC protocol and to LowMC. As a consequence, it is an order of magnitude faster than Picnic2. See [KZ20] for more details. The performance numbers we provide are for Picnic3.

LowMC

Both Picnic and Picnic2 require a one-way function at their core: this function must of course be hard to invert, but should have as few multiplications as possible, since these are costly to evaluate in MPC. Thus they use LowMC [ARS+15], a block cipher tailored for MPC and designed to have a very small multiplicative complexity. Although recent, LowMC has been studied in a fair amount of works [DKP+19, JNRV20, LIM20].

Multi-Target Attack on Round 1 Picnic

A recent attack [DN19] has shown that in the Round 1 version of Picnic, an adversary knowing Q signatures could recover the private key about $128 \cdot Q$ times faster than predicted by the theory. This attack has been mitigated in the Round 2 submission.

NIST level	|SK| (bytes)	|PK| (bytes)	|sig| (bytes)	KG (cycles)	Sign (cycles)	Verify (cycles)
1	17	34	12359	4151	18252055	13811201
3	24	48	27173	6567	37595772	29243365
5	32	64	46282	9504	65555710	46887830

8 SPHINCS+ (alternate)

Type: Signature
Paradigm: Hash-then-sign
Family: One-way 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]

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] to protect 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] for an up-to-date presentation of the framework.

Variants

SPHINCS+ admits several variants: in addition to the 3 security levels (128, 192 or 256), there are 3 choices for the underlying building block (SHAKE-256, SHA-256 or Haraka). Additionally, one could choose between a "small" and a "fast" variant: these provide a trade-off between the size of the signature and the running time of the signing procedure. Finally, one could choose between a "simple" variant which is simpler and faster, and a "robust" variant which has a more conservative security argument. Hence there are $3 \times 3 \times 2 \times 2 = 36$ variants. The performance numbers provided here are those of 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].

Physical Attacks

While no black-box attack has been proposed against SPHINCS+, a side-channel attack [KGB+18] has shown how an unprotected implementation can leak part of the private key. Similarly, [CMP18] showed theoretically that one single, mildly-controlled fault injection can lead to the recovery of the private key, and [GKPM18] carried this attack on an ARM Cortex M3.

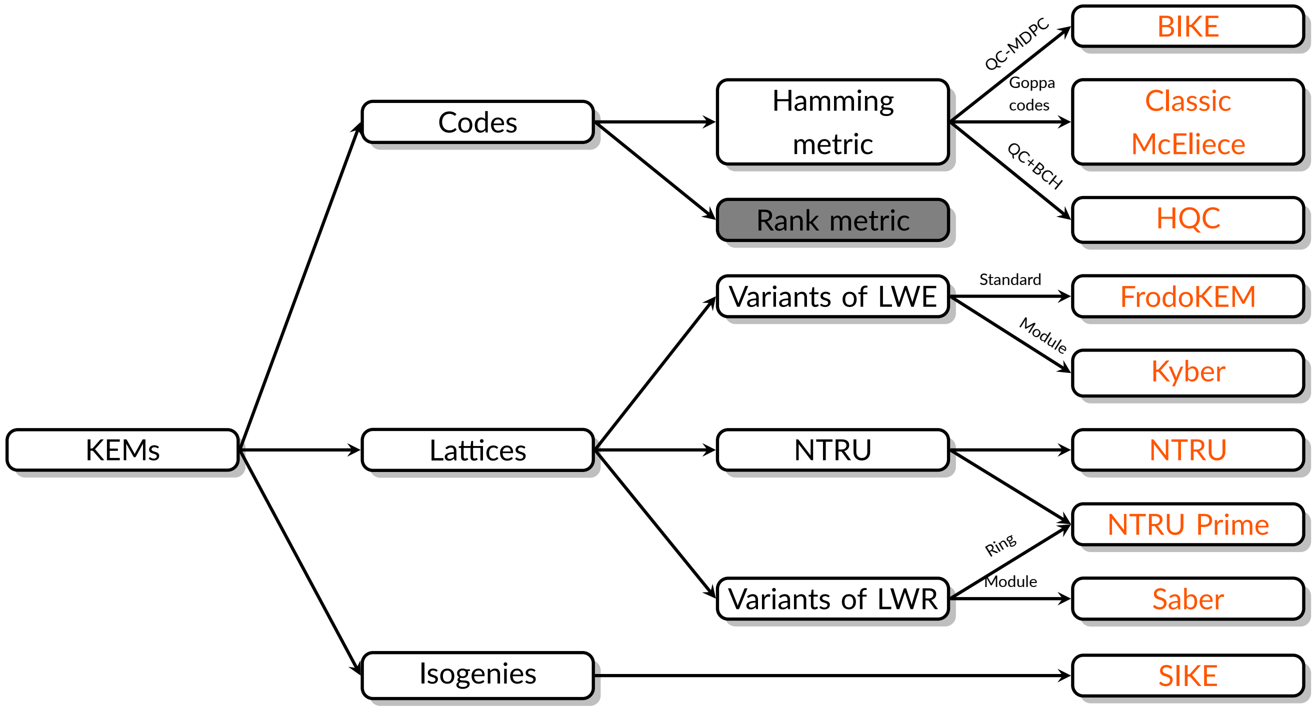
NIST level	|SK| (bytes)	|PK| (bytes)	|sig| (bytes)	KG (cycles)	Sign (cycles)	Verify (cycles)
1	64	32	17088	2748026	68541826	4801338
3	96	48	35664	4063066	113484456	7552358
5	128	64	49856	21327470	435984168	14938510

9 Key-Encapsulation Mechanisms

We now study the key-establishment schemes under consideration for standardization. In Round 1 of NIST's standardization process (December 2017), 49 submissions for key-establishment were accepted [NIS17b]. After a preliminary analysis by the cryptographic community, NIST selected 17 of these 49 submissions for Round 2 of the standardization process [NIS19a]. Out of these 17 submissions, 9 were selected to Round 3 of the standardization process [NIS20b]: 4 *finalists* and 5 *alternates*.

- Classic McEliece (finalist);
- Kyber (finalist);
- NTRU (finalist);
- Saber (finalist);
- BIKE (alternate);
- FrodoKEM (alternate);
- HQC (alternate);
- NTRU Prime (alternate);
- SIKE (alternate).

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 a CCA transform to a base key-exchange/encryption scheme, and therefore usually claims the best security guarantees (security against active attackers). Thus, for simplicity we will only consider KEMs. An overview of the KEMs can be found in the figure below.



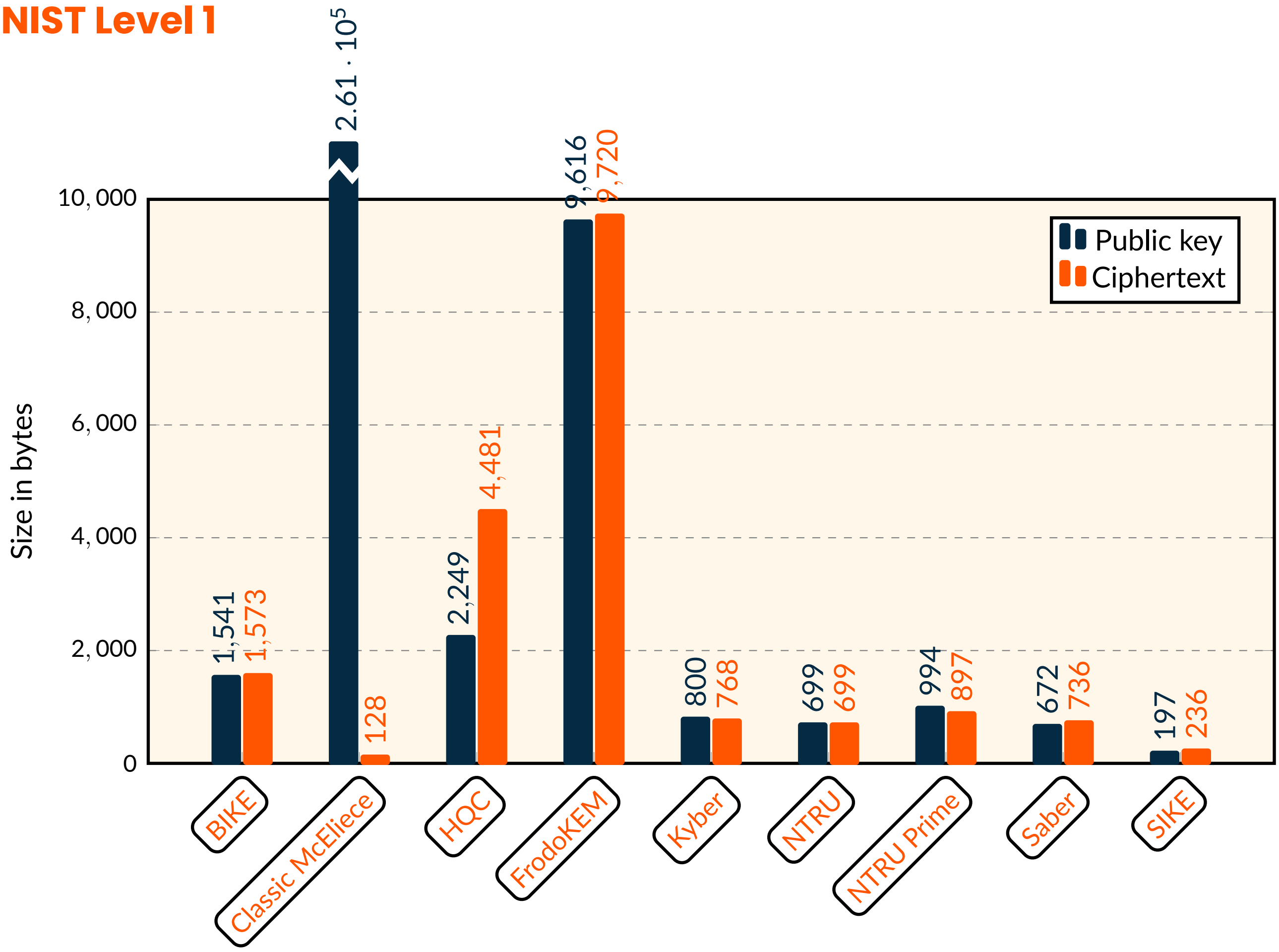
In addition, we provide the following data:

- A comparative performance study of the 9 KEMs (pages 18 to 21);
- For each scheme, one page summarizing its main facts (pages 23 to 31).

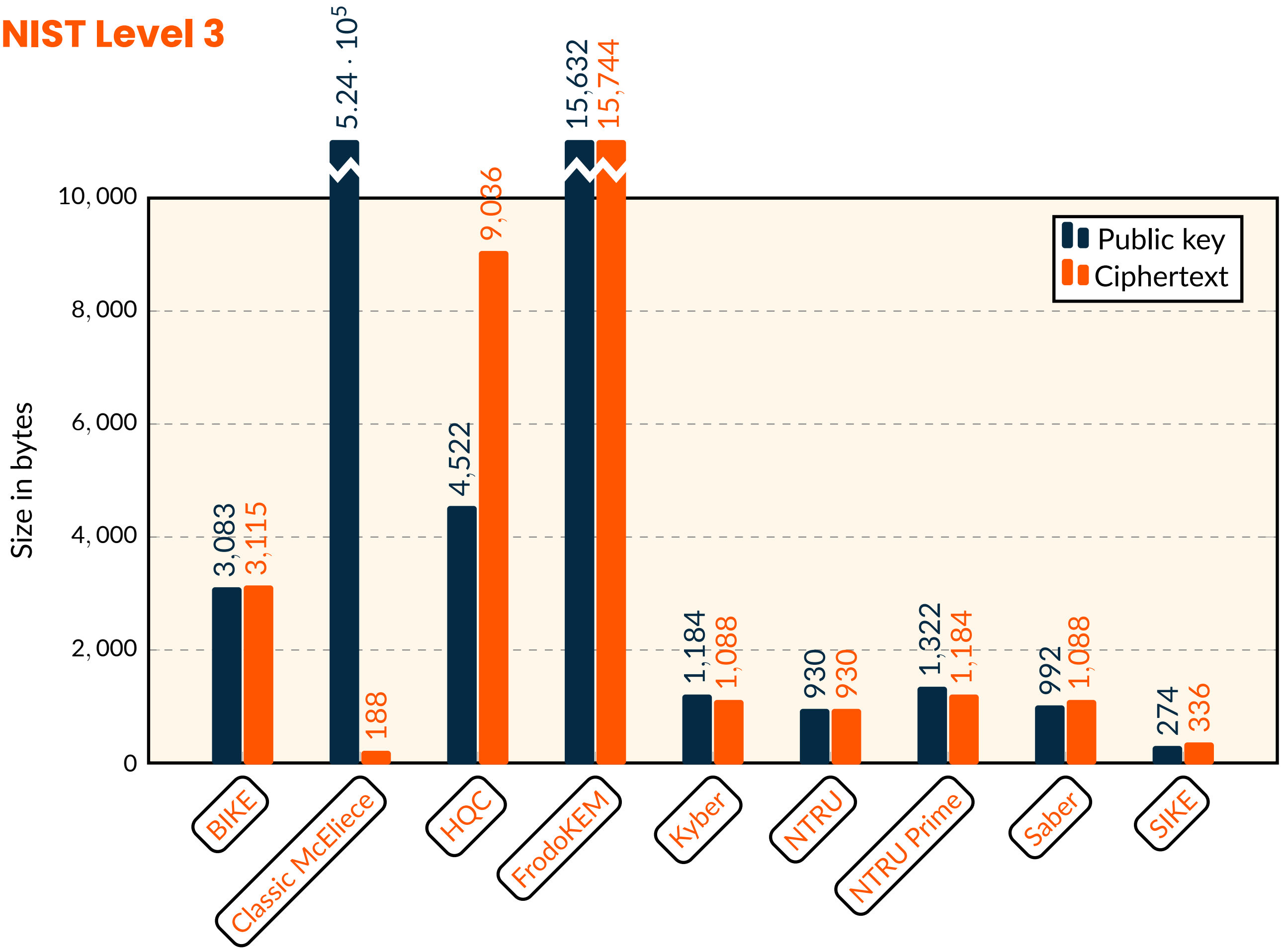
Communication Costs

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.

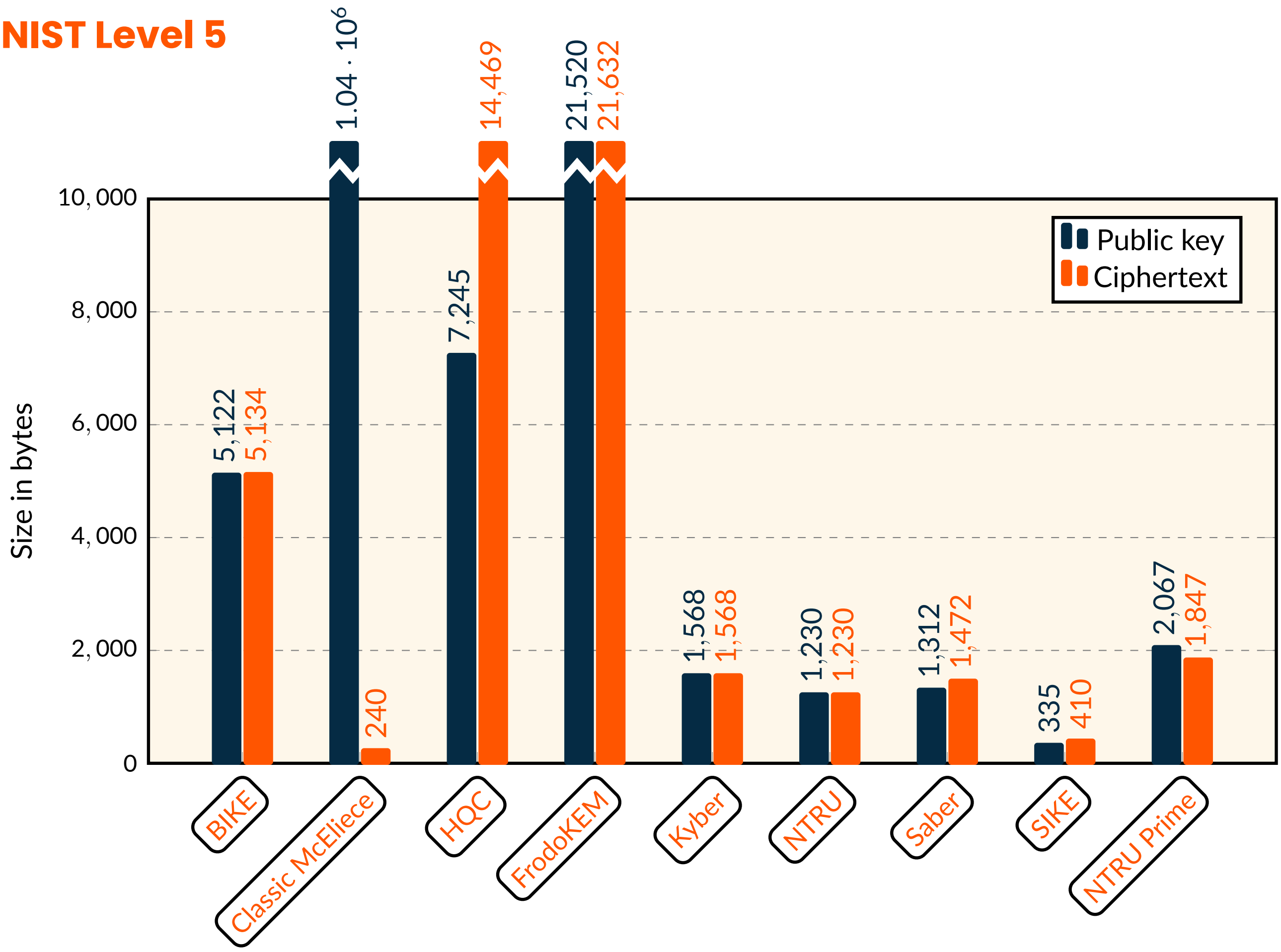
NIST Level 1



NIST Level 3



NIST Level 5

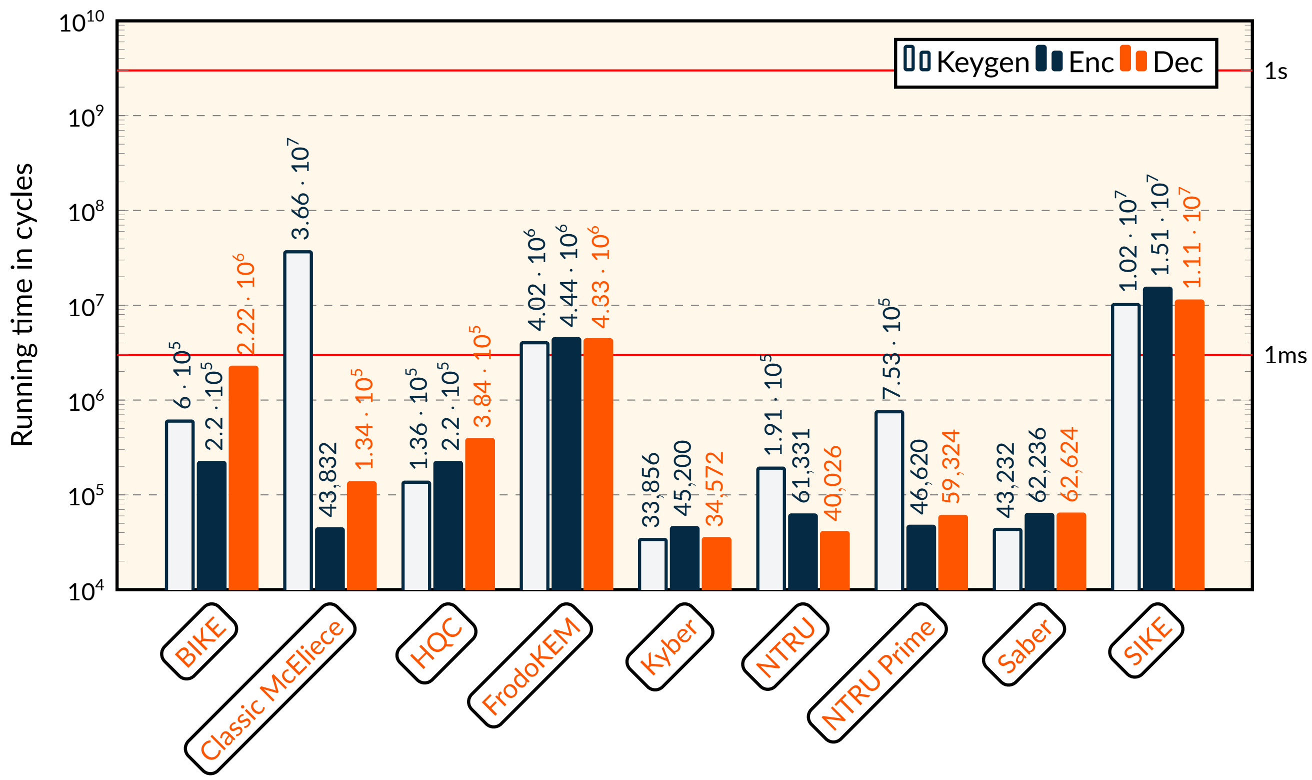


Computational Costs

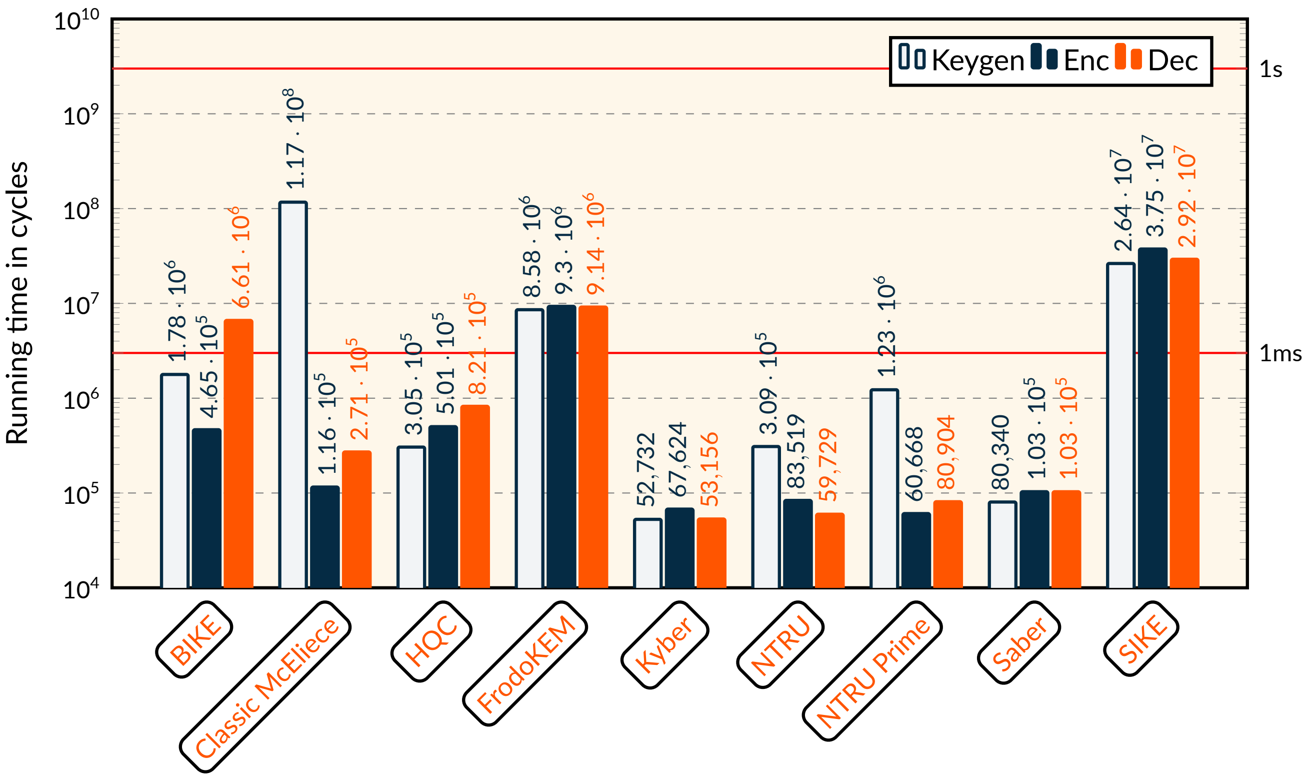
We now compare the running times in cycles of the 9 KEMs, for optimized implementations targeting x64 platforms. All numbers are extracted from the specification documents of the schemes (which might be inaccurate) and were obtained on different platforms. Therefore, they may not enable a completely fair comparison. To make these numbers less abstract, each graph also contains two horizontal red lines that correspond respectively to 1 millisecond and 1 second 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 200 times faster than FrodoKEM, and about 600 times faster than SIKE. Again, these numbers do not tell the full story, as FrodoKEM relies on hardness assumptions that are far less structured than Kyber's, and SIKE is much cheaper in terms of bandwidth.

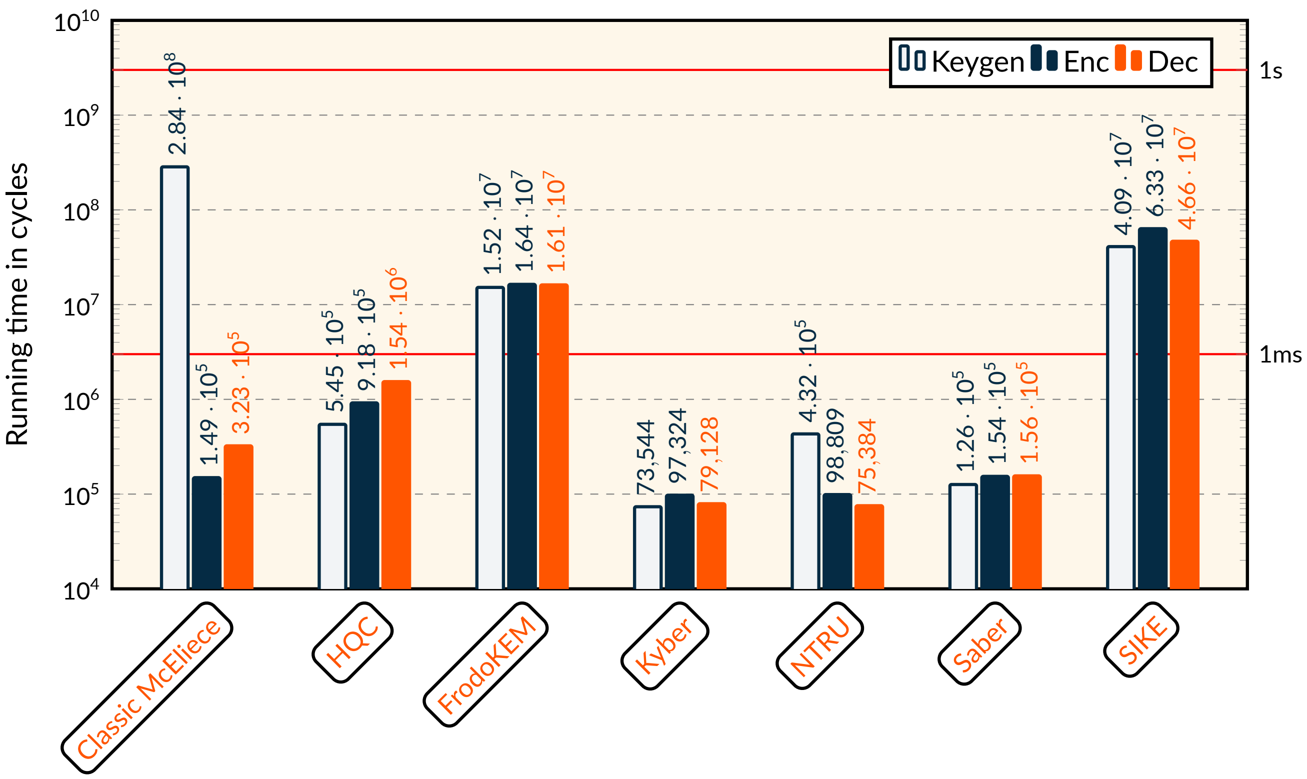
NIST Level 1



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.

10 Classic McEliece (finalist)

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]

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

Design-wise, the most notable novelty of Classic McEliece is perhaps the CCA transform that is used to obtain an IND-CCA KEM from a OW-CPA public-key encryption scheme. 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 demonstrated 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

11 Kyber (finalist)

Type: KEM
CCA Transform: Tweaked Fujisaki-Okamoto [FO99] with implicit rejection [HHK17]
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]

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 goes as follows:

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

Encryption goes as follows:

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

Decryption goes as follows:

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

Module Lattices

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

Hashing the Public Key

Kyber achieves CCA security by performing a variant of Fujisaki-Okamoto's transform. One interesting fact is that 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. D'Anvers pointed out in [NIS17a] that this technique could invalidate Kyber's security proof (though in practice, it does not seem to introduce a concrete weakness). Consequently, the compression technique was removed in the Round 2 specification.

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

12 NTRU (finalist)

Type: KEM (and Encryption)
CCA Transform: $U_m^{\not\perp}$ [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]

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 makes it easy to implement in constant time and eliminates decryption failures [HNP+03], "evaluate-at-1" attacks and invertibility checks. There are several ways to interpret and prove the CCA transform used by NTRU: either as the $U_m^{\not\perp}$ transform of [HHK17], the one of [SXY18] or the SimpleKEM transform from [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], this time 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

13 Saber (finalist)

Type: KEM
CCA Transform: $FO^{\not\perp}$ transform [HHK17, JZC+18]
Family: Lattices
Hard Problems: Module-LWR (Learning With Rounding)
Sym. primitives: SHA-3, SHAKE-128
Randomness: Uniform
Specification: [DKR+20]
Website: `https://www.esat.kuleuven.be/cosic/pqcrypto/saber/`
Related Works: [DKRV18, JZC+18]

High-Level Design

Just like Kyber, Saber is based on the Lindner-Peikert framework. The main difference is that it uses Module-LWR instead of Module-LWE: the "random noise" is replaced with "deterministic rounding", making the implementation simpler but changing the underlying hardness assumption. The conversion into a IND-CCA scheme is done via the $FO^{\not\perp}$ transform [HHK17, JZC+18]. An early version [DKRV18] of Saber relied on *Noisy Diffie-Hellman* key-exchange, but its design has switched to LPR-style [LPR10] encryption since then.

A Simple Design

Saber makes several design choices oriented at simplicity. Like Kyber, it only works with elements over $\mathbb{Z}_q[x]/(x^{256}+1)$. Moreover, the use of LWR simplifies its description. Finally, the integer modulus q is taken to be a power of two. As a consequence of this choice of q, polynomial multiplications are done via the Toom-Cook and Karatsuba algorithms.

Implementations

Saber has been implemented on several platforms. The specification document reports implementations on ARM Cortex-M4, HW/SW codesigns and/or complete hardware implementations on Artix-7 and Virtex Ultrascale+, a masked implementation on ARM Cortex-M4 and even an implementation on the RSA coprocessor ESP32 (inspired from a similar result [AHH+18] for Kyber).

NIST level	|SK| (bytes)	|PK| (bytes)	|ctxt| (bytes)	KG (cycles)	Enc (cycles)	Dec (cycles)
1	1568	672	736	43232	62236	62624
3	2304	992	1088	80340	103204	103092
5	3040	1312	1472	126220	153832	155700

14 BIKE (alternate)

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	$s \leftarrow c_0 h_0 + c_1 h_1$; $u \leftarrow 0$	$s \leftarrow ch_0$; $u \leftarrow 0$	$s \leftarrow c_0 + c_1 h_0$; $u \leftarrow t/2$
	$(e_0', e_1') \leftarrow \texttt{Decode}(s, h_0, h_1, u)$		
	$K \leftarrow \mathbf{K}(e_0', e_1')$		

Design and Variants

BIKE is based on QC-MDPC codes – this acronym stands for *Quasi-Cyclic Moderate Density Parity Check*. The quasi-cyclicity allows dramatic gains in compactness and speed. BIKE originally came in three variants, BIKE-{1,2,3}, presented in the above table extracted from the Round 1 presentation of BIKE. In a simplification effort, only BIKE-2 was kept in the last iteration. Note that BIKE-2 was the most compact of the three variants; it also used to have the slowest key generation procedure, but this was recently mitigated 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	-	-	-

15 HQC (alternate)

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: [MAB+20]
Website: `http://pqc-hqc.org`
Related Works: [Ale03, Gab05, ABD+16a, DGZ17]

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

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 Kyber, Saber, FrodoKEM and NTRU Prime (in its LPRime variant). 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.

Hardware Implementation

The specification document of HQC gives performance numbers for an implementation on the Artix-7 FPGA.

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

16 FrodoKEM (alternate)

Type: KEM (and Encryption)
CCA Transform: Variant of $\mathsf{FO}_m^{\not\perp}$ [HHK17, JZC+18]
Family: Lattices
Hard Problems: LWE
Sym. primitives: SHAKE-128/256 (and optionally AES-128/256)
Randomness: Centered rounded Gaussians (using a CDT)
Specification: [NAB+20]
Website: `https://frodokem.org/`
Related Works: [LP11, BCD+16, ATT+18, BFM+18, HOKG18, BFM+19]

Standard Lattices

FrodoKEM is an instantiation of the Lindner-Peikert framework [LP11] (see Kyber). It is the only remaining scheme that works over unstructured lattices. These lattices entail working with matrices having entries in $\mathbb{Z}$, whereas most other lattice-based candidates take entries in $\mathbb{Z}_q[x]/(f)$ for some polynomial f. As a result, FrodoKEM has larger communication costs, but relies on more conservative (or at least less structured) hardness assumptions.

FrodoCCS

FrodoKEM is an evolution of FrodoCCS [BCD+16]. FrodoCCS was based on *Noisy Diffie-Hellman*, but FrodoKEM switched in favor of public key encryption. In addition, they use different error distributions and symmetric primitives.

Implementations

Despite its somewhat large memory footprint, many works studied the implementation of FrodoKEM on embedded devices. For example, [HOKG18] proposes designs for FPGA and microprocessors, with the goal to minimize area consumption on FPGAs (resp. peak stack usage on microprocessors). This is achieved mainly by using memory-laziness tricks. [BFM+18] informally argue that the generation of a very large public matrix (A) doesn't require a cryptographically secure PRNG, and uses a PRNG known as xoshiro** for this step.

Side-Channel Attacks

[ATT+18] studied horizontal attacks against FrodoCCS, and [BFM+19] studied single-trace attacks against FrodoKEM. A timing attack [GJN20] was recently displayed against the reference implementation of Round 2 FrodoKEM, which performed comparison (during decapsulation) in variable time. This has been fixed in Round 3. The flaw was also present in LAC, BIKE, HQC, ROLLO , RQC .

NIST level	|SK| (bytes)	|PK| (bytes)	|ctxt| (bytes)	KG (cycles)	Enc (cycles)	Dec (cycles)
1	19888	9616	9720	4022000	4440000	4325000
3	31296	15632	15744	8579000	9302000	9143000
5	43088	21520	21632	15191000	16357000	16148000

17 NTRU Prime (alternate)

Type: KEM
CCA Transform: Variant of Dent [Den03] with confirmation hash
Family: Lattices
Hard Problems: NTRU, Ring-LWR
Sym. primitives: SHA-512, AES-256-CTR
Randomness: Uniform in $\{-1, 1\}$ or $\{-1, 0, 1\}$, sometimes with bounded weight
Specification: [BBC+20]
Website: `https://ntruprime.cr.yp.to/`
Related Works: [Den03, BCLv17]

Design

The high-level design of NTRU Prime was introduced in [BCLv17]. Just like NTRU, it tweaks the original NTRU encryption scheme [HPS98] in order to make it easily implementable in constant time, and to eliminate decryption failures [HNP+03], "evaluate-at-1" attacks and invertibility checks.

Reducing the Attack Surface

A point emphasized by the specification of NTRU Prime is the choice of the base field, which is $\mathbb{Z}_q[x]/(x^p - x - 1)$, where p, q are two primes such that $(x^p - x - 1)$ is prime in $\mathbb{Z}_q[x]$. The explicit goal behind this choice is to hedge against future cryptanalytic attacks similar to those exploiting the presence of subfields [ABD16b, BBdV+17] or computable homomorphisms [CGS14, CDPR16], while keeping the efficiency gains provided by a ring structure.

Polynomial Multiplication

The rings $\mathbb{Z}_q[x]/(x^p - x - 1)$ chosen by NTRU Prime do not natively support the number theoretic transform (NTT). Until recently, NTRU Prime used Karatsuba and Toom-Cook for multiplying polynomials, but recent progress [ACC+21] has made the NTT competitive for NTRU Prime rings.

Two Variants

The NTRU Prime submission proposes two variants: Streamlined NTRU Prime, and NTRU LPRime. At a high level, the first variant is close to NTRU, but low-level details differ significantly, see [SHRS17, Section 6.1] and [Sch18]. The second variant instantiates the Lindner-Peikert framework (see Kyber) using a variant of Ring-LWR over the NTRU Prime ring. The performance numbers we provide are for Streamlined NTRU Prime.

Embedded Implementations

The specification and official website of NTRU Prime report optimized implementations on ARM Cortex-M4 and AVR ATmega1284 microcontrollers, and Xilinx Zynq Ultrascale+ FPGA.

NIST level	|SK| (bytes)	|PK| (bytes)	|ctxt| (bytes)	KG (cycles)	Enc (cycles)	Dec (cycles)
1	1518	994	897	752904	46620	59324
3	1999	1322	1184	1227380	60668	80904
5	3059	2067	1847	-	-	-

18 SIKE (alternate)

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]

History and design

SIKE is the only isogeny-based candidate scheme. At a high level, it implements the SIDH key-exchange [JD11]. Unlike the classical Diffie-Hellman, it is not fully interactive. An attack in [GPST16] can be adapted to break the CCA security of the basic SIDH design. Hence SIKE uses a conversion inspired from [HHK17] to ensure IND-CCA security. 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.

Cryptanalysis

The current best attack against SIKE is 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.

Implementation Flaws

Two flaws in the reference implementation of SIKE has been recently unearthed [NIS19b]. The comparison step in the re-encryption part of the decapsulation procedure was improperly computed, seemingly voiding out the CCA security claim. This has been subsequently corrected.

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

References

[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 Zemor, Valentin Vasseur, and Santosh Ghosh. BIKE. Technical report, 2020. available at [NIS20a].

[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, 2020. available at [NIS20a].

[ABD+16a] 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. http://eprint.iacr.org/2016/1194.

[ABD16b] Martin R. Albrecht, Shi Bai, and Léo Ducas. A Subfield Lattice Attack on Overstretched NTRU Assumptions - Cryptanalysis of Some FHE and Graded Encoding Schemes. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 153–178. Springer, Heidelberg, August 2016.

[ACC+21] Erdem Alkim, Dean Yun-Li Cheng, Chi-Ming Marvin Chung, Hülya Evkan, Leo Wei-Lun Huang, Vincent Hwang, Ching-Lin Trista Li, Ruben Niederhagen, Cheng-Jhih Shih, Julian Wälde, and Bo-Yin Yang. Polynomial Multiplication in NTRU Prime. *IACR TCHES*, 2021(1):217–238, 2021. https://tches.iacr.org/index.php/TCHES/article/view/8733.

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

[AHH+18] Martin R. Albrecht, Christian Hanser, Andrea Hoeller, Thomas Pöppelmann, Fernando Virdia, and Andreas Wallner. Implementing RLWE-based Schemes Using an RSA Co-Processor. *IACR TCHES*, 2019(1):169–208, 2018. https://tches.iacr.org/index.php/TCHES/article/view/7338.

[Ale03] Michael Alekhnovich. More on Average Case vs Approximation Complexity. In *44th FOCS*, pages 298–307. IEEE Computer Society Press, October 2003.

[ARS+15] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. Ciphers for MPC and FHE. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT 2015, Part I*, volume 9056 of *LNCS*, pages 430–454. Springer, Heidelberg, April 2015.

[ATT+18] Aydin Aysu, Youssef Tobah, Mohit Tiwari, Andreas Gerstlauer, and Michael Orshansky. Horizontal side-channel vulnerabilities of post-quantum key exchange protocols. In *2018 IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2018, Washington, DC, USA, April 30 - May 4, 2018*, pages 81–88. IEEE Computer Society, 2018.

[BBC+20] Daniel J. Bernstein, Billy Bob Brumley, Ming-Shing Chen, Chitchanok Chuengsatiansup, Tanja Lange, Adrian Marotzke, Bo-Yuan Peng, Nicola Tuveri, Christine van Vredendaal, and Bo-Yin Yang. NTRU Prime. Technical report, 2020. available at [NIS20a].

[BBdV+17] Jens Bauch, Daniel J. Bernstein, Henry de Valence, Tanja Lange, and Christine van Vredendaal. Short Generators Without Quantum Computers: The Case of Multiquadratics. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 27–59. Springer, Heidelberg, April / May 2017.

[BCD+16] Joppe W. Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the Ring! Practical, Quantum-Secure Key Exchange from LWE. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016*, pages 1006–1018. ACM Press, October 2016.

[BCLv17] Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. NTRU Prime: Reducing Attack Surface at Low Cost. In Carlisle Adams and Jan Camenisch, editors, *SAC 2017*, volume 10719 of *LNCS*, pages 235–260. Springer, Heidelberg, August 2017.

[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.

[Beu20] Ward Beullens. Improved Cryptanalysis of UOV and Rainbow. Cryptology ePrint Archive, Report 2020/1343, 2020. https://eprint.iacr.org/2020/1343.

[BFM+18] Joppe W. Bos, Simon Friedberger, Marco Martinoli, Elisabeth Oswald, and Martijn Stam. Fly, you fool! Faster Frodo for the ARM Cortex-M4. Cryptology ePrint Archive, Report 2018/1116, 2018. https://eprint.iacr.org/2018/1116.

[BFM+19] Joppe W. Bos, Simon Friedberger, Marco Martinoli, Elisabeth Oswald, and Martijn Stam. Assessing the Feasibility of Single Trace Power Analysis of Frodo. In Carlos Cid and Michael J. Jacobson Jr:, editors, *SAC 2018*, volume 11349 of *LNCS*, pages 216–234. Springer, Heidelberg, August 2019.

[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.

[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.

[CDG+17] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. Post-Quantum Zero-Knowledge and Signatures from Symmetric-Key Primitives. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1825–1842. ACM Press, October / November 2017.

[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. NTRUEncrypt. Technical report, 2020. available at [NIS20a].

[CDPR16] Ronald Cramer, Léo Ducas, Chris Peikert, and Oded Regev. Recovering Short Generators of Principal Ideals in Cyclotomic Rings. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 559–585. Springer, Heidelberg, May 2016.

[CFM+20] A. Casanova, J.-C. Faugère, G. Macario-Rat, J. Patarin, L. Perret, and J. Ryckeghem. GeMSS. Technical report, 2020. available at [NIS20a].

[CGS14] Peter Campbell, Michael Groves, and Dan Shepherd. Soliloquy: A Cautionary Tale, 2014. https://docbox.etsi.org/workshop/2014/201410_CRYPTO/S07_Systems_and_Attacks/S07_Groves_Annex.pdf.

[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.

[DCP+20] Jintai Ding, Ming-Shing Chen, Albrecht Petzoldt, Dieter Schmidt, Bo-Yin Yang, Matthias Kannwischer, and Jacques Patarin. Rainbow. Technical report, 2020. available at [NIS20a].

[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.

[DKP+19] Itai Dinur, Daniel Kales, Angela Promitzer, Sebastian Ramacher, and Christian Rechberger. Linear Equivalence of Block Ciphers with Partial Non-Linear Layers: Application to LowMC. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part I*, volume 11476 of *LNCS*, pages 343–372. Springer, Heidelberg, May 2019.

[DKR+20] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Frederik Vercauteren, Jose Maria Bermudo Mera, Michiel Van Beirendonck, and Andrea Basso. SABER. Technical report, 2020. available at [NIS20a].

[DKRV18] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. Saber: Module-LWR Based Key Exchange, CPA-Secure Encryption and CCA-Secure KEM. In Antoine Joux, Abderrahmane Nitaj, and Tajjeeddine Rachidi, editors, *AFRICACRYPT 18*, volume 10831 of *LNCS*, pages 282–305. Springer, Heidelberg, May 2018.

[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.

[DN19] Itai Dinur and Niv Nadler. Multi-target Attacks on the Picnic Signature Scheme and Related Protocols. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 699–727. Springer, Heidelberg, May 2019.

[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.

[DS05] Jintai Ding and Dieter Schmidt. Rainbow, a New Multivariable Polynomial Signature Scheme. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *ACNS 05*, volume 3531 of *LNCS*, pages 164–175. Springer, Heidelberg, June 2005.

[DY13] Jintai Ding and Bo-Yin Yang. Degree of Regularity for HFEv and HFEv-. In Philippe Gaborit, editor, *Post-Quantum Cryptography - 5th International Workshop, PQCrypto 2013*, pages 52–66. Springer, Heidelberg, June 2013.

[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.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure Integration of Asymmetric and Symmetric Encryption Schemes. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 537–554. Springer, Heidelberg, August 1999.

[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.

[GJN20] Qian Guo, Thomas Johansson, and Alexander Nilsson. A Key-Recovery Timing Attack on Post-quantum Primitives Using the Fujisaki-Okamoto Transformation and Its Application on FrodoKEM. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 359–386. Springer, Heidelberg, August 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.

[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.

[GPST16] Steven D. Galbraith, Christophe Petit, Barak Shani, and Yan Bo Ti. On the Security of Supersingular Isogeny Cryptosystems. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part I*, volume 10031 of *LNCS*, pages 63–91. Springer, Heidelberg, December 2016.

[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, 2020. available at [NIS20a].

[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.

[HOKG18] James Howe, Tobias Oder, Markus Krausz, and Tim Güneysu. Standard Lattice-Based Key Encapsulation on Embedded Devices. *IACR TCHES*, 2018(3):372–393, 2018. https://tches.iacr.org/index.php/TCHES/article/view/7279.

[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. `http://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, 2020. available at [NIS20a].

[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.

[JNRV20] Samuel Jaques, Michael Naehrig, Martin Roetteler, and Fernando Virdia. Implementing Grover Oracles for Quantum Key Search on AES and LowMC. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 280–310. Springer, Heidelberg, May 2020.

[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.

[JZC+18] Haodong Jiang, Zhenfeng Zhang, Long Chen, Hong Wang, and Zhi Ma. IND-CCA-Secure Key Encapsulation Mechanism in the Quantum Random Oracle Model, Revisited. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 96–125. Springer, Heidelberg, August 2018.

[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.

[KKW18] Jonathan Katz, Vladimir Kolesnikov, and Xiao Wang. Improved Non-Interactive Zero Knowledge with Applications to Post-Quantum Signatures. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *ACM CCS 2018*, pages 525–537. ACM Press, October 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.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced Oil and Vinegar Signature Schemes. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 206–222. Springer, Heidelberg, May 1999.

[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/.

[KZ20] Daniel Kales and Greg Zaverucha. Improving the Performance of the Picnic Signature Scheme. *IACR TCHES*, 2020(4):154–188, 2020. https://tches.iacr.org/index.php/TCHES/article/view/8680.

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

[LDK+20] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. CRYSTALS-DILITHIUM. Technical report, 2020. available at [NIS20a].

[LIM20] Fukang Liu, Takanori Isobe, and Willi Meier. Cryptanalysis of Full LowMC and LowMC-M with Algebraic Techniques. Cryptology ePrint Archive, Report 2020/1034, 2020. https://eprint.iacr.org/2020/1034.

[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.

[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.

[MAB+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, 2020. available at [NIS20a].

[McE78] Robert J. McEliece. A Public-Key Cryptosystem Based on Algebraic Coding Theory. *JPL DSN Progress Report*, 44, 05 1978.

[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. http://eprint.iacr.org/2012/409.

[NAB+20] Michael Naehrig, Erdem Alkim, Joppe Bos, Leo Ducas, Karen Easterbrook, Brian LaMacchia, Patrick Longa, Ilya Mironov, Valeria Nikolaenko, Christopher Peikert, Ananth Raghunathan, and Douglas Stebila. FrodoKEM. Technical report, 2020. available at [NIS20a].

[NIE86] H. NIEDERREITER. Knapsack-type cryptosystems and algebraic coding theory. *Prob. Control and Inf. 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.

[NIS17a] NIST. Official Comments (Round 1) - Kyber, 2017. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/official-comments/CRYSTALS-KYBER-official-comment.pdf.

[NIS17b] NIST. Post-Quantum Cryptography - Round 1 Submissions, 2017. https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

[NIS19a] NIST. NISTIR 8240 - Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process, 2019. https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8240.pdf.

[NIS19b] NIST. Official Comments (Round 3) - qTESLA, 2019. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-3/official-comments/SIKE-round3-official-comment.pdf.

[NIS20a] Post-Quantum Cryptography - Round 3 Submissions, 2020. https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions.

[NIS20b] NIST. NISTIR 8309 - Status Report on the Secind Round of the NIST Post-Quantum Cryptography Standardization Process, 2020. https://csrc.nist.gov/publications/detail/nistir/8309/final.

[NIW+20] Shuhei Nakamura, Yasuhiko Ikematsu, Yacheng Wang, Jintai Ding, and Tsuyoshi Takagi. New Complexity Estimation on the Rainbow-Band-Separation Attack. Cryptology ePrint Archive, Report 2020/703, 2020. https://eprint.iacr.org/2020/703.

[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.

[Pat96] Jacques Patarin. Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 33–48. Springer, Heidelberg, May 1996.

[PBB10] Albrecht Petzoldt, Stanislav Bulygin, and Johannes Buchmann. CyclicRainbow - A Multivariate Signature Scheme with a Partially Cyclic Public Key. In Guang Gong and Kishan Chand Gupta, editors, *INDOCRYPT 2010*, volume 6498 of *LNCS*, pages 33–48. Springer, Heidelberg, December 2010.

[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.

[PCG01] Jacques Patarin, Nicolas Courtois, and Louis Goubin. QUARTZ, 128–Bit Long Digital Signatures. In David Naccache, editor, *CT-RSA 2001*, volume 2020 of *LNCS*, pages 282–297. Springer, Heidelberg, April 2001.

[Pet20] Albrecht Petzoldt. Efficient Key Generation for Rainbow. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, pages 92–107. Springer, Heidelberg, 2020.

[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, 2020. available at [NIS20a].

[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.

[PQS20] PQShield. An Overview of Post-Quantum Cryptography. 2020. https://pqshield.com/whitepapers/.

[PS20] Ray Perlner and Daniel Smith-Tone. Rainbow Band Separation is Better than we Thought. Cryptology ePrint Archive, Report 2020/702, 2020. https://eprint.iacr.org/2020/702.

[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, Leo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, and Damien Stehle. CRYSTALS-KYBER. Technical report, 2020. available at [NIS20a].

[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.

[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. http://eprint.iacr.org/2013/004.

[SSH11] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. On Provable Security of UOV and HFE Signature Schemes against Chosen-Message Attack. In Bo-Yin Yang, editor, *Post-Quantum Cryptography - 4th International Workshop, PQCrypto 2011*, pages 68–82. Springer, Heidelberg, November / December 2011.

[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.

[TPD20] Chengdong Tao, Albrecht Petzoldt, and Jintai Ding. Improved Key Recovery of the HFEv- Signature Scheme. Cryptology ePrint Archive, Report 2020/1424, 2020. https://eprint.iacr.org/2020/1424.

[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.

[ZCD+19] Greg Zaverucha, Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, Jonathan Katz, Xiao Wang, and Vladmir Kolesnikov. Picnic. Technical report, National Institute of Standards and Technology, 2019. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions.

[ZCD+20] Greg Zaverucha, Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, Jonathan Katz, Xiao Wang, Vladmir Kolesnikov, and Daniel Kales. Picnic. Technical report, 2020. available at [NIS20a].

[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.