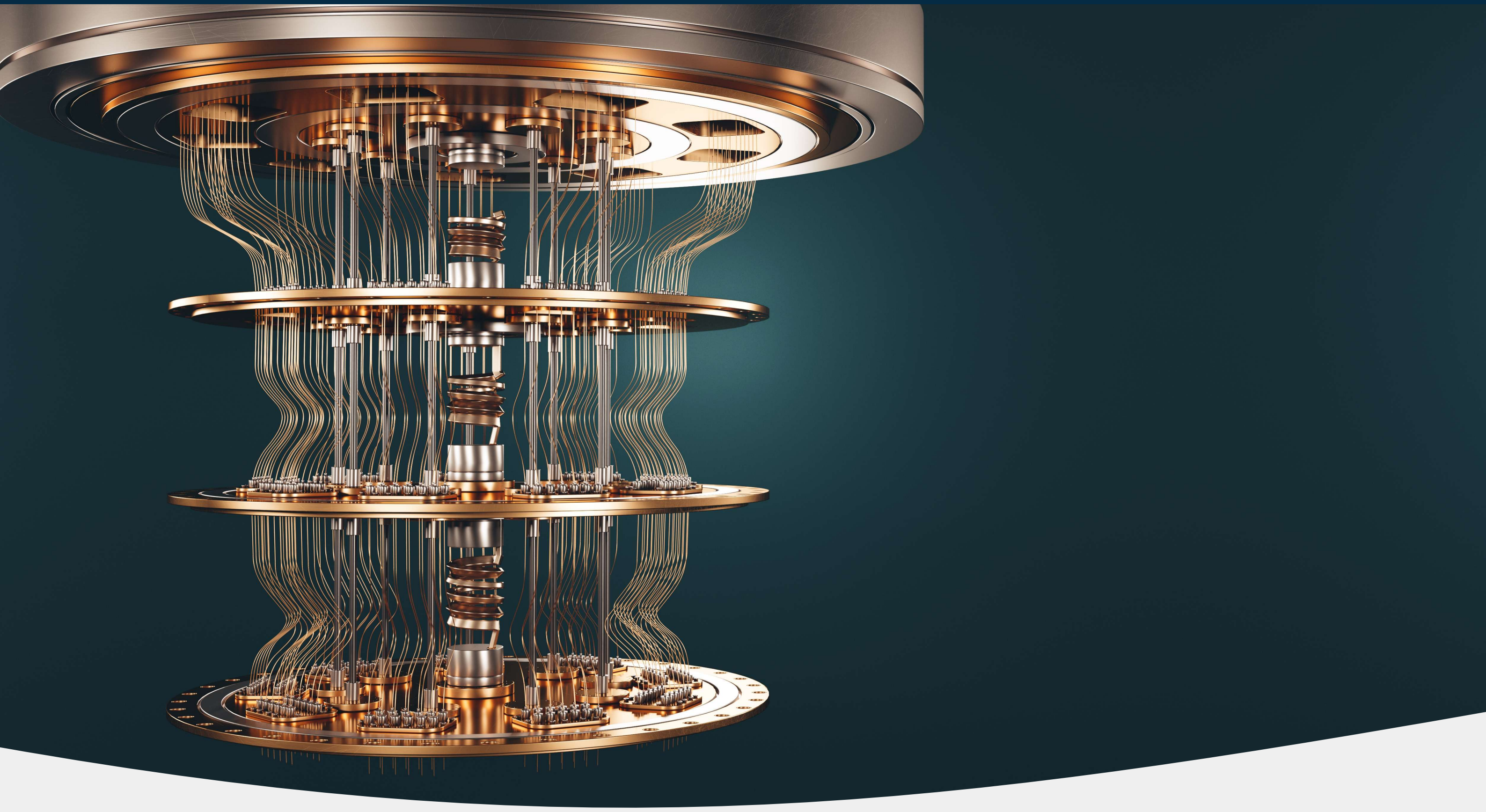
Think openly, build securely



White paper:

An Overview of Post-Quantum Cryptography

PQShield

February 10, 2021

Contents

This document is an introduction to post-quantum cryptography. While there exists a large corpus of literature on the topic, the field has not yet coalesced into a mature and well-structured discipline. As a result, even people familiar with it may struggle to get a clear understanding of the technical principles and how they are put into practice.

The goal of this document is to provide the reader with an understanding of the key technical ideas used in post-quantum cryptography. As such, it is a rather technical document. However, we focus here on the high-level principles, and try to avoid low-level details when possible. An interesting fact is that subfields of post-quantum cryptography (lattice-based, code-based, multivariate, etc.) often share several techniques and design principles, even though they may work on very different mathematical objects. We highlighted these connections when we felt they were relevant.

1 Classical Public-Key Cryptography

Cryptography deals with the issue of ensuring secure communications over insecure channels, and does so via mathematical methods.

When two parties share a secret key, symmetric-key cryptography provides efficient ways of ensuring the confidentiality (via symmetric encryption, for example AES [AES01]) and integrity of communications (via message authentication codes, for example HMAC [HMA08]).

However, these solutions are inapplicable when the parties *do not* share a secret key in advance. Public-key cryptography was invented precisely to deal with these situations.

In this section, we first describe some hard problems, the building blocks upon which public-key cryptography is based today. We then explain the key technical ideas, and show how cryptographic protocols put them into practice. Finally, we explain how quantum computers jeopardize the security provided by current public-key cryptography.

1.1 Hard problems

Computationally hard problems, or hard problems for short, is a broad notion encompassing problems that require a significant (ideally, intractable) amount of resources to be solved. Cryptography makes a peculiar use of hard problems; other fields often try to avoid these problems, but cryptography uses them as the foundation of secure schemes. This is typically done by establishing an equivalence between the security of a scheme and the intractability of a hard problem.

Until recently, two hard problems (or variants thereof) have been ubiquitous in public-key cryptography: *integer factorisation*, and the *discrete logarithm problem*.

Factorisation-related Problems

The prime factorisation problem is one of the simplest, and probably best-known, hard problems in cryptography.

The Prime Factorisation Problem

Let p and q be two prime integers and $N = p \times q$. Given N, find p and q.

Whether this problem is actually hard (with the computational power currently available) depends on the set from which p and q are picked. For actual cryptosystems, p and q are picked from a set large enough so that prime factorisation is infeasible in practice.

While a few cryptosystems [Rab79, Wil84] rely solely on prime factorisation, many more of them [GM82, Pai99] rely on related problems, such as the RSA problem.

The RSA Problem

Let p and q be prime integers, $N = p \times q$, and e and d two integers such that

$$d \times e = 1 \bmod (p-1) \times (q-1)$$

Given N, e and $m^e \bmod N$ for a random $0 \leq m < N$, find m.

The hardness of the RSA problem is the assumption underlying the security of the eponymous encryption and signature schemes by Rivest, Shamir and Adleman [RSA78].

The prime factorisation problem is *at least as hard* as the RSA problem, however whether they are equivalent is still an open question. In practice, schemes based on the RSA problem choose their parameters to make the related factorisation problem hard.

Problems related to the Discrete Logarithm

Another class of hard problems heavily relied upon by cryptography relates to the discrete logarithm in finite cyclic groups. A finite group is a finite set with some added algebraic structure: for example, the set $\mathbb{Z}_q$ of integers modulo q is a finite group. We say that a finite group $G = \langle g \rangle$ is cyclic if it is generated by an element g. We first consider the discrete logarithm problem, or DLOG.

DLOG - Discrete Logarithm Problem

Let $G = \langle g \rangle$ be a finite cyclic group. Given g and g^a, find a.

Similarly to prime factorisation and RSA, while some schemes rely solely on DLOG, others rely on two related problems: DDH and CDH, first used by Diffie and Hellman [DH76].

DDH and CDH - Diffie-Hellman Problems

Let $G = \langle g \rangle$ be a finite group. Given g, g^a, g^b for a, b random:
Decision (DDH): Distinguish (g^a, g^b, g^{ab}) from a triple (g^a, g^b, g^c) with c random.
Search (CDH): Compute g^{ab}.

DLOG is at least as hard as CDH, which is at least as hard as DDH. In other words:

$$DLOG \geq CDH \geq DDH.$$

Assessing equivalence in general is still complicated; there are groups in which CDH is as hard as DLOG, and there are groups for which DDH is easy but CDH seems hard.

An active subfield of public-key cryptography is elliptic-curve cryptography. An elliptic curve is essentially the set of points (x, y) verifying a certain equation for fixed a, b, p:

$$y^2 = x^3 + ax + b \bmod p.$$

Taking G to be an elliptic curve often allows to design very compact schemes based on the problems DLOG, CDH, DDH, etc.

1.2 Encryption, Key Exchange and Key Encapsulation

Encryption schemes, key-exchange protocols and key encapsulation mechanisms are three related protocols which solve the same problem: establishing a secure communication between two parties over an insecure channel.

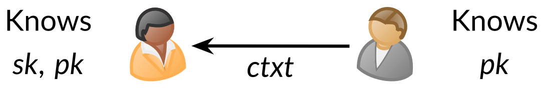
- A key-exchange protocol is a (possibly interactive) protocol at the end of which two parties agree on a shared symmetric key.
- A (public-key) encryption scheme is a scheme with public and private keys, where a public key *pk* allows to encrypt a message *msg*, and the corresponding private key *sk* allows to recover *msg*.
- A key encapsulation mechanism (or KEM) also has public and private keys, a public key *pk* allows to encrypt a random symmetric key K (*encapsulation*), and the associated private key *sk* allows to recover K (*decapsulation*).

Most basic constructions achieve either encryption schemes [RSA78, ElG85] or key-exchange protocols [DH76] naturally, but there are simple generic conversions that transform any protocol of one type into a protocol of an other type.

Encryption

For simplicity, each time we describe a scheme, Alice will denote the owner of the private key *sk* (for decryption) and Bob the owner of the corresponding public key *pk* (for encryption).

In an encryption scheme, Alice ( ) generates a key pair (sk, pk), keeps the private key sk to herself, and publicly distributes the public key pk. Bob ( ) computes a ciphertext $ctxt = \mathsf{Enc}(pk, msg)$.



Upon reception of $ctxt$, Alice applies a decryption algorithm to it and recovers the message: $msg = \mathsf{Dec}(sk, ctxt)$. Alice and Bob now both know a secret msg.

Given $\mathsf{Enc}(pk, msg)$, it must be hard to recover msg, except if we know the private key sk, which allows to invert $\mathsf{Enc}(pk, \cdot)$. We have:

$$\mathsf{Dec}(sk, \mathsf{Enc}(pk, msg)) = msg.$$

This description is arguably very generic. We illustrate how to apply this idea in practice with a simplified version of the RSA cryptosystem, described hereafter. The idea is that while exponentiating a message ($ctxt = msg^e \bmod N$) is easy, "inverting" this operation to recover msg is hard if the RSA problem is hard.

RSA Encryption

- The private key sk is a couple (p, q) of distinct primes and a value d;
- The public key pk is the product $N = p \times q$, and a value e such that

$$e \times d = 1 \bmod (p-1) \times (q-1);$$

- To encrypt a message msg, compute:

$$ctxt = msg^e \bmod N$$

- To decrypt $ctxt$, compute:

$$msg = ctxt^d \bmod N$$

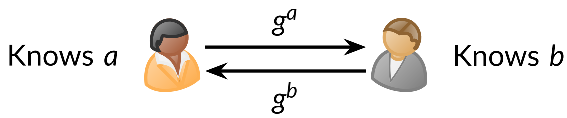
A "magic" mathematical property of d and e is that, for any element x in $\mathbb{Z}_N$:

$$(x^e)^d = x^{ed} = x \bmod N.$$

This allows Alice (who knows d) to recover msg. Of course, this simplistic description conceals a number of details which have to be addressed in real applications. This is the purpose of transforms such as RSA-OAEP [BR95].

Diffie-Hellman Key Exchange

The Diffie-Hellman protocol is a key-exchange protocol. Given a random (generating) element g of a group G, Alice chooses a random a, computes g^a, keeps a to herself and send g^a to Bob. Similarly, Bob sends some g^b to Alice. Upon reception of g^b, Alice raises it to the power of her secret exponent a, and obtains $(g^b)^a$. Bob does the same, and obtains $(g^b)^a$.



Thanks to the properties of the exponentiation, both obtain the same value:

$$(g^a)^b = (g^b)^a = g^{ab}$$

For someone observing the exchange, finding g^{ab} is as hard as solving CDH, which is conjectured to be hard.

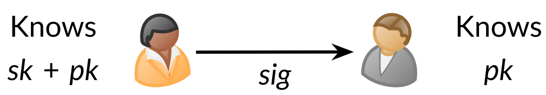
This protocol verifies some nice additional properties: it is *static* (Alice can reuse her secret value a for an exchange with another party), and *non-interactive* (both parties can send their g^x before receiving an answer). We will later see that post-quantum protocols struggle to achieve both properties at once.

Chosen-Ciphertext Security

Most basic encryption schemes are only proven secure against chosen-plaintext attacks (CPA), where an attacker must encrypt honestly. In chosen-ciphertext attacks (CCA), an attacker can craft malicious ciphertexts. These can be mitigated by applying generic conversions, the most famous being the Fujisaki-Okamoto transforms [FO99a, FO99b].

1.3 Signatures

Digital signatures solve the same problem as their physical counterparts: proving the authenticity of a document. (sk, pk) will denote a pair of signing and verification keys, and H a cryptographic hash function. Alice attests the validity of a document to Bob by appending a digital signature $sig \leftarrow \text{Sign}(sk, msg)$ to it. sig guarantees that Alice is the rightful author of the document, and detects any modification.



The Hash-then-Sign Paradigm

The Hash-then-sign paradigm is arguably the most "intuitive" approach for building digital signatures. The message is first hashed by a hash function H into a challenge c. A signature is a value sig such that $f_{pk}(sig) = \text{c}$ for some public function f_{pk} parameterized by pk. f_{pk} is a trapdoor one-way function, which means it is easy to compute and hard to invert (*one-way*), except if we know sk (the *trapdoor*).

With the RSA problem, this idea is easily put in application, as described below. In practice, executing this idea securely requires adddi-tional tweaks described in RSA-PSS [BR98].

RSA Signature

- The private key sk is a couple (p, q) of distinct primes and a value d;
- The public key pk is the product $N = p \times q$, and a value e such that

$$e \times d = 1 \bmod (p-1) \times (q-1);$$

- To sign a message msg, compute:

$$sig = H(msg)^d \bmod N$$

- A signature is accepted if and only if:

$$H(msg) = sig^e \bmod N$$

The Fiat-Shamir Paradigm

The philosophy of the Fiat-Shamir paradigm [FS87] is very different from Hash-then-sign: it is based on *identification protocols*. These protocols are *interactive* protocols that allow a prover to prove its identity to a verifier. Most of them follow a *commit-challenge-response* communication flow illustrated in Figure 1.

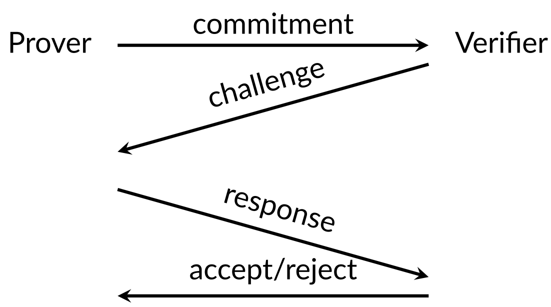


Figure 1: A 3-pass identification protocol

The most important phase is the response, where the prover solves a problem dependent on both the commitment and the challenge, and which would be hard to solve without their private key. The Fiat-Shamir paradigm turns an identification protocol into a signature scheme by making the challenge a hash of both the commitment and the message. Of course, the resulting scheme is now non-interactive.

Schnorr signature

- The private key sk is a value x;
- The public key pk is $h = g^x$;
- To sign a message msg:
  - **Commit:** Select a random y, and compute $u = g^y$;
  - **Challenge:** Compute $c = H(u, msg)$;
  - **Response:** z = y - cx;

  The signature is $sig = (u, z)$.
- A signature is accepted if and only if:

$$u = g^z h^{H(u,msg)}.$$

An illustration of this principle is the Schnorr signature scheme [Sch90] described above, where each of Alice's signatures is a proof that she knows the discrete logarithm of the public key.

1.4 The Impact of Quantum Computers

Most of the cryptographic schemes based on the prime factorisation, RSA, discrete logarithm and Diffie-Hellman problems would be assumed secure if not for quantum computers.

In 1994, Shor [Sho94] showed that these classically hard problems would be *easy* to solve on a large scale quantum computer. Since quantum computers were mostly theoretical objects at the time, there was no immediate impact.

However, there has been significant progress on building these computers, driven by large companies (Microsoft, IBM, etc.) and state actors (China, USA, EU), and the prospect of a practical quantum computer becomes more tangible every year.

This has led the cryptographic community, the industry and many standards bodies to plan a replacement of today's widely used public-key cryptography by a quantum-safe alternative: *post-quantum cryptography*.

Summary	
Inception:	1976
Hard Problems:	RSA, Factoring, CDH, DDH
Enc/KEM:	(elliptic-curve) Diffie-Hellman, RSA Encryption
Signatures:	Schnorr, RSA signatures

2 Lattice-based Cryptography

A lattice is a set generated by integer linear combinations of the columns of a matrix. Thus, lattice-based cryptosystems and hard problems typically involve matrices. Lattice-based cryptography is rather recent (1996) compared to the other subfields, but it has seen a steady growth since its inception.

With a few exceptions, lattice-based cryptography started with very theoretical constructions targeting provable security. As it stands, several schemes are proven secure under the hardness of various lattice problems. However, not all proofs are equal, in the sense that some proofs have a limited practical relevance [CKMS16].

Today, there exist several cryptographic constructions based on lattices. Beyond encryption and signatures, more advanced constructions have been proposed, such as homomorphic encryption, identity-based encryption, etc.

The efficiency of cryptographic schemes based on generic lattices is moderate. Many schemes rely on more structured lattices, and achieve high efficiency in the process. This comes at the cost of more aggressive hardness assumptions.

2.1 Hard Problems

There is a myriad of conjectured hard problems in lattice-based cryptography. The most common are SIS and LWE. Both work with matrices having their entries in a finite ring $\mathcal{R}$ (for example $\mathbb{Z}_q$ or $\mathbb{Z}_q[x]/(x^d + 1)$).

SIS - Short Integer Solution

Given $A \in \mathcal{R}^{n \times m}$, find a short vector $v \neq 0$ such that $Av = 0$.

Solving SIS without the shortness constraint is straightforward linear algebra. However, forcing the solution to be short adds a geometric constraint which makes this problem much harder.

Another widespread hard problem is *Learning With Errors*, or LWE for short. The definition of LWE gives cryptosystems' designers the freedom to choose not only $\mathcal{R}, m, n$, but also the secret distribution, error distribution, etc. The secret and error distributions (which are public) typically have a small support.

LWE - Learning With Errors

Let $A \in \mathcal{R}^{n \times m}$ be a uniformly random and $b = A^t s + e$, where $s \in \mathcal{R}^n$ and $e \in \mathcal{R}^m$ are vectors sampled from the *'secret' distribution* and *'error' distribution*, respectively.

Decision: Distinguish (A, b) from values sampled uniformly.

Search: Given (A, b), find s.

This versatility is a double-edged sword. Along with the rich algebraic structure of LWE, it allows to build several constructions on LWE beyond signatures and key agreement, the most pre-eminent being *homomorphic encryption*, which enables secure computation over encrypted data. However, this also leaves room for cryptosystems' designers to unwittingly use insecure parameters.

The most prevalent flavors of LWE are standard LWE, *module*-LWE (or MLWE) and *ring*-LWE (or RLWE). Standard LWE takes $\mathcal{R} = \mathbb{Z}_q$, the integers modulo q. MLWE takes $\mathcal{R}$ to be some polynomial ring, for example

$\mathcal{R} = \mathbb{Z}_q[x]/(x^d + 1)$. RLWE is the special case of MLWE where $n = 1$. MLWE and RLWE introduce some structure which can be exploited to have more efficient schemes (faster and more compact). However, this added structure also means that the underlying hardness assumptions are more aggressive. SIS also exists in similar flavors.

As an illustration, RLWE with one sample posits that:

$$(a, a * s + e) \in \mathcal{R}^2$$

is hard to distinguish from uniform, where a, s, and e are sampled from some uniform, secret and error distributions over $\mathcal{R}$, respectively. Because of its simplicity, we will take RLWE to illustrate many of our examples.

2.2 Key Exchange and Encryption

2.2.1 Encryption

RLWE Encryption

- The private key contains a, s, e;
- The public key is $(a, b = a * s + e)$;
- The encryption of *msg* is $ctxt = (u, v)$, with:

$$u = r * a + e_1$$
$$v = r * b + e_2 + \left\lceil \frac{q}{2} \right\rfloor msg$$

  where e_1, e_2 are error vectors.
- To decrypt a ciphertext *ctxt*, the owner of the private key computes:

$$v - u * s,$$

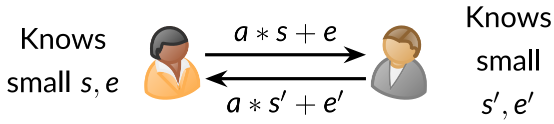
  which is equal to $\left\lceil \frac{q}{2} \right\rfloor$ *msg* plus some noise. As the noise is small (thus concentrated in the low bits) and *msg* is encoded in the high bits, it is easy to recover *msg*.

There are several approaches for building encryption schemes using lattices. The box "RLWE Encryption" above presents a simplified version (using RLWE) of the most common approach [LPR10, LP11].

Several schemes are based on this framework, including candidates for standardization FrodoKEM [NAB+17], Kyber [SAB+17] and Saber [DKRV17].

2.2.2 Noisy Diffie-Hellman

One could imagine a naive adaptation of the Diffie-Hellman key-exchange using RLWE. Here, with a being a public element:



However, this would give only an *approximately* shared secret since:

$$a * s * s' + s' * e \neq a * s * s' + s * e'.$$

To cope with this issue, the notion of *reconciliation* has been introduced [DXL12, Pei14]. The idea is that one of the parties sends a *hint* to the other party, so that they agree on the same shared secret. This is known as *noisy Diffie-Hellman*.

Unfortunately, this approach lacks a few advantages of the classical Diffie-Hellman. It is interactive and cannot be used with static keys (e.g. Alice cannot use s, e twice). A more mundane issue is the presence of a patent on reconciliation [Din12].

2.3 Signatures

Secure digital signatures using lattices have not been immediate to obtain. The first secure proposals (for both the hash-then-sign and Fiat-Shamir paradigms) were designed in 2009.

2.3.1 The GPV Framework

Early attempts [GGH97, HHP^{+}03] to adapt the hash-then-sign paradigm to the lattice setting took the public key *pk* and private key *sk* to be, respectively, a long basis and a short basis of a same lattice. The signing procedure does the following:

1. Hash the message into a point $H(msg)$ of the ambient space of the lattice.
2. Use *sk* to find a lattice point *v* close to $H(msg)$.

The verification procedure checks that *v* is close to $H(msg)$, and uses *pk* to check that it is also in the lattice. As performing step 2. is hard to do with a short basis but easy with a long basis, it was expected that breaking this signature scheme would not be easier in practice than solving hard lattice problems. This idea is illustrated in Figure 2; one can see that the signature lies at the intersection of the lattice and of a parallelepiped generated by the private key and centered at $H(msg)$, which makes it unique.

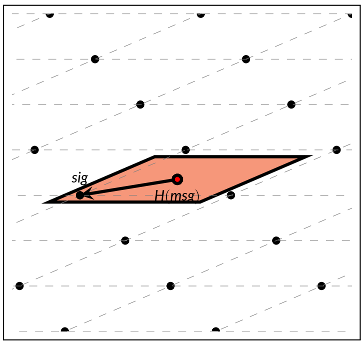


Figure 2: Hash-then-sign *à la* [GGH97, HPS98]. Signatures leak the private key.

However, if step 2. is not done properly, each signature will lie in a parallelepiped having the form of the private key, and as such would leak a little bit of information about its geometry. This was exploited in devastating attacks [NR06, DN12]; in which a few thousand (msg, sig) pairs leaked enough information to fully recover the private key.

A countermeasure was proposed by Gentry, Peikert and Vailuntanathan in 2008 [GPV08]. The idea is to use a randomized variant of the algorithm used in step 2. of the original idea [GGH97, HPS98]. This randomization ensures that one message has several valid signatures, and is done in a way which eliminates any correlation between the secret key and the distribution of the signatures. In addition to thwarting the attacks of [GGH97, HPS98], this actually makes the framework of [GPV08] provably secure under the hardness of standard lattice problems.

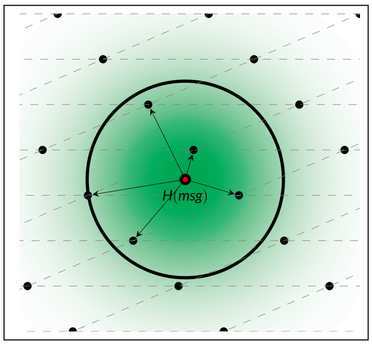


Figure 3: Hash-then-sign *à la* [GPV08]. Signatures no longer leak the private key.

Falcon is an application of the GPV framework. This technique can also be used to build more advanced primitives such as (hierarchical) identity-based encryption [GPV08, CHKP10, DLP14], attribute-based encryption [Boy13, BGG^{+}14], etc.

2.3.2 Fiat-Shamir with Aborts

As with their hash-then-sign counterparts, initial attempts [HPS01] to build lattice signatures with the Fiat-Shamir paradigm were quickly cryptanalysed [GJSS01, GS02]. Like for hash-then-sign schemes, each signature leaked a small part of the private key.

A provably secure mitigation was proposed by Lyubashevsky [Lyu09]. The issue with previous attempts was that the underlying protocol lacked the *zero-knowledge* property, and the point of failure was in the *response* step of the protocol. In the attacked schemes, this step induced a distribution of the signatures correlated to the private key, hence the attacks. This weakness did not appear for classical Fiat-Shamir schemes.

Lyubashevsky observed that randomly aborting (and starting over) the protocol, in a carefully chosen way, allowed to *eliminate* the correlation of the signatures with the private key and nullified this class of attacks. This addition enabled the zero-knowledge property, which was the only property missing in order to make the lattice-based schemes based on Fiat-Shamir provably secure. This technique, by Lyubashevsky, is called *Fiat-Shamir with Aborts*.

Fiat-Shamir with Aborts (using LWE)

- The private key sk is two matrices S, E of small norm.
- The public key is $(\mathrm{A}, \mathrm{T} = \mathrm{A} \times \mathrm{S} + \mathrm{E})$.
- To sign a message msg:
  - **Commit:** Generate small random vectors $\mathrm{y}_1, \mathrm{y}_2$ and compute the commitment $\mathrm{u} = \mathrm{A} \times \mathrm{y}_1 + \mathrm{y}_2$;
  - **Challenge:** Compute $\mathrm{c} = H(\mathrm{u}, msg)$;
  - **Response:** Compute $\mathrm{z}_1 = \mathrm{y}_1 + \mathrm{S} \times \mathrm{c}$ and $\mathrm{z}_2 = \mathrm{y}_2 + \mathrm{E} \times \mathrm{c}$;
  - **Abort:** With a certain probability $p(sig, sk)$, restart;

  The signature is $sig = (\mathrm{c}, \mathrm{z}_1, \mathrm{z}_2)$.
- A signature is accepted if and only if:
  - $(\mathrm{z}_1, \mathrm{z}_2)$ is short;
  - $\mathrm{c} = H(\mathrm{A} \times \mathrm{z}_1 + \mathrm{z}_2 - \mathrm{T} \times \mathrm{c}, msg)$.

One can see that this scheme is surprisingly similar to Schnorr signatures described in Section 1.3: the public element $a \in G$ is now a matrix A, the challenge $u = g^y$ is replaced with $\mathrm{u} = \mathrm{A} \times \mathrm{y}_1 + \mathrm{y}_2$, and the underlying hard problem DLOG has been replaced with LWE.

We note one important addition; the *abort* step is added as to avoid any leakage of the key and make the scheme secure. Schemes based on Fiat-Shamir with aborts include Dilithium [LDK+19].

Summary

Inception:	1996
Assumptions:	LWE (*Learning with Errors*), SIS (*Short Integer Solution*), NTRU
Enc/KEM:	FrodoKEM, Kyber, Saber
Signatures:	Dilithium, Falcon, qTESLA

3 Code-based Cryptography

Error-correcting codes usually serve to guarantee the integrity and reliability of communication over unreliable channels, by detecting and removing errors. Code-based cryptography uses them in a completely different way, by deliberately adding errors to the point that removing them is hard, except for someone who knows a secret description of the code. Since it mostly entails simple algebraic operations (Gaussian elimination, multiplication, sometimes inversion) on finite field elements, code-based cryptography is often amenable to fast hardware implementations.

Code-based cryptography was first introduced by McEliece [McE78] in his eponymous encryption scheme. His original scheme remains fundamentally secure, is reasonably fast and has short ciphertexts, but a very large public key. Many attempts have been made at making it more efficient, but doing so in a secure manner has proven to be delicate.

Achieving secure code-based signatures has been an even more difficult task. Novel proposals in this direction are being made, but only the test of time will determine if these efforts are successful.

3.1 Problems

Code-based cryptography relies on linear error-correcting codes (or codes for short). These are generated by matrices over finite fields (for simplicity, this exposition will focus on the field $\mathbb{F}_2$, the integers modulo 2). The generating matrix of a code C is often denoted G, and we often associate to it a parity-check matrix of C, denoted by H and which verifies $G \times H^t = 0$.

Syndrome Decoding

The most common code-based problem is the syndrome decoding problem.

Syndrome Decoding

Given a matrix $H \in \mathbb{F}_2^{k \times n}$ and a *syndrome* $s \in \mathbb{F}_2^k$, find $e \in \mathbb{F}_2^n$ of Hamming weight at most t such that $H \times e = s$.

The syndrome decoding problem is very similar to the SIS problem described in Section 2.1: the matrix $A \in \mathcal{R}^{n \times m}$ is replaced by $H \in \mathbb{F}_2^{k \times n}$, and the constraint on the norm of the solution is replaced by a constraint on its Hamming weight. In a certain parameter regime (not used in cryptography), syndrome decoding is a NP-complete problem.

One can come up with variations of the syndrome decoding problem. One such variation is the rank syndrome decoding problem, which replaces $\mathbb{F}_2$ by a different field, and imposes a condition on the *rank* of the solution instead of its Hamming weight. The rank metric family of schemes [AGH+17, AAB+19, ABD+19] are based on this variation. Another popular modification is to impose a (quasi-)cyclic structure, like schemes based on QC-MDPC codes do.

Code Indistinguishability

Another hard problem, or rather a family of hard problems, informally states that for a given family $\mathcal{C} = \{C_i\}_i$ of codes, it is hard to distinguish a random matrix G generating a code $C = C_i$ from a random matrix.

At a high level, it is somewhat similar to the EIP problem described in Section 4.1. The similarity does not stop here; while this problem is presumed hard for some families of codes (e.g. Goppa, QC-MDPC and LRPC codes), many other choices (Reed-Solomon, LDPC, etc.) have led to insecure schemes.

3.2 Encryption

The idea underlying code-based encryption schemes is that a message will be encrypted by adding some noise to it, and that removing this noise is hard except if one knows a secret description of some code.

McEliece

A common way of using codes for encryption comes from McEliece's original scheme.

The hardness of syndrome decoding and of distinguishing $\hat{G}$ from a random matrix is necessary for McEliece's encryption scheme to be secure.

McEliece Encryption

- The private key contains a permutation P, an invertible matrix S, and a matrix G generating a linear code C capable of correcting t errors. The algorithm for correcting errors using G is public
- The public key is the product $\hat{G} = SGP$.
- The encryption of a message *msg* is:

  $$ctxt = msg \times \hat{G} + z,$$

  with z a vector of Hamming weight t.
- To decrypt a ciphertext *ctxt*, the owner of the private key computes:

  $$ctxt \times P^{-1} = msg \times SG + z \times P^{-1}.$$

  Since the error vector $z \times P^{-1}$ has weight t, the matrix G can be used to decode the above value to $msg \times S$, and then multiplying this by S^{-1} yields the plaintext *msg*.

McEliece's original proposal has large public keys, since these are large matrices. The use of structured matrices allows to diminish this size, though this automatically implies a more aggressive hardness assumption.

Niederreiter

McEliece's cryptosystem admits a *dual* version which was proposed by Niederreiter in 1986 [NIE86]. It replaces the generator matrix G by an associated parity check matrix H.

Niederreiter Encryption

- The private key contains P, S and G as for McEliece's cryptosystem.
- The public key is $H_{pk} = SHP$, for H the parity check matrix of C.
- The encryption of a message *msg* is:

  $$ctxt = H_{pk} \times msg,$$

  msg being encoded as an error vector containing at most t ones.
- The decryption procedure computes:

  $$S^{-1} \times ctxt = HP \times msg,$$

  then a syndrome decoding algorithm followed by linear algebra is applied to recover *msg*.

Another difference is that in McEliece's scheme, an error is added to the encoded message, whereas here the message itself becomes the error. Niederreiter's scheme also provides some trade-offs in term of sizes and speed. Security-wise, both schemes are strictly equivalent. Schemes based on Niederreiter's scheme include standard candidates Classic McEliece and BIKE.

3.3 Signatures

Obtaining secure and efficient signatures from error-correcting codes has been very hard to achieve so far. Several attempts have failed. We present here two secure but currently inefficient signature schemes, and two new proposals.

Hash-then-sign

In 2001, Courtois, Finiasz and Sendrier [CFS01] proposed a signature scheme based on the Hash-then-sign paradigm. It converts Niederreiter's scheme into a signature scheme: decryption becomes the signing procedure, and encryption becomes the verification procedure. Unfortunately, the parameters of [CFS01] do not scale well, which yields impractical parameters.

A recent construction [DST19] revisited the [CFS01] scheme and proposed to apply the GPV framework (see Section 2.3) to build code-based signatures. It is provably secure under a *new* code-based assumption.

Fiat-Shamir

Stern [Ste94, Ste96] and Véron [Vér96] proposed identification schemes from error-correcting codes. In this setting, the secret key is a vector e, and the public key is a random matrix H as well as the syndrome $s = H \times e$. The prover proves knowledge of e in a zero-knowledge way. Unfortunately, these protocols have soundness error between $2/3$ and $1/2$, so converting them into signature schemes via the Fiat-Shamir transform would require a few hundred repetitions. This would result in a slow signing procedure and large signatures.

A recent signature scheme, Durandal [ABG+19], uses partially *Fiat-Shamir with aborts* (see Section 2.3) in conjunction with the rank metric. It achieves a much better soundness than Stern-like protocols, but does not have a full security proof.

Summary	
Inception:	1978
Assumptions:	Syndrome decoding, Distinguishing from a random code
Enc/KEM:	BIKE, Classic McEliece, HQC, NTS-KEM, ROLLO
Signatures:	CFS, Durandal, WAVE

4 Multivariate Cryptography

Multivariate cryptography is rather old since its inception dates back to 1988, when the first multivariate encryption scheme was proposed. It builds cryptosystems from problems involving multivariate polynomial equations over finite fields, for example the $\mathcal{MQ}$ problem.

However, over the years many schemes have been broken; while this may seem paradoxical, it is easily explained by the fact that these schemes relied on other, less secure and sometimes implicit assumptions. This has arguably undermined the credibility of the field.

The landscape of multivariate signatures is more mature than that of their encryption counterparts. There exist both hash-then-sign and Fiat-Shamir signatures with solid foundations, even though their concrete security for practical parameters can still be hard to pinpoint.

Multivariate hash-then-sign schemes usually have small signatures (a few hundred bytes) at the expense of large keys. For Fiat-Shamir signatures, it is the other way around.

4.1 Hard Problems

Multivariate cryptography is based on multivariate polynomial equations. For example:

$$f(x_1, x_2) = 3x_1^3x_2 + x_1^2 - x_2^3 + x_2 + 1$$

is a multivariate polynomial (of degree 4, since $x_1^3x_2$ is of degree 4). Solving problems involving multivariate polynomials of degree > 1 is conjectured hard for sufficiently large parameters. The variables are usually in a finite field (for example, $\mathbb{Z}_q$ for some prime q).

The $\mathcal{MQ}$ Problem

The most famous problem is $\mathcal{MQ}$; it entails working with multivariate quadratics (i.e. multivariate polynomials of degree at most two), hence its name.

The $\mathcal{MQ}$ problem

Let $\mathbb{F}$ be a finite field. Let $\mathsf{F}(\mathsf{x})$ be $(f_1(\mathsf{x}), \ldots, f_m(\mathsf{x}))$, where each $f_i : \mathbb{F}^n \to \mathbb{F}$ is a multivariate polynomial of degree at most 2 in $\mathsf{x} = (x_1, \ldots, x_n)$. Let $\mathsf{y} \in \mathbb{F}^m$ and F be inputs to the problems below.

Decision: Is there x such that $\mathsf{F}(\mathsf{x}) = \mathsf{y}$?
Search: Find x such that $\mathsf{F}(\mathsf{x}) = \mathsf{y}$.

The decision version of $\mathcal{MQ}$ is NP-complete [GJ79], and its search version is NP-hard. This makes $\mathcal{MQ}$ an attractive option for building cryptographic schemes, but we will see that putting this idea into practice has been difficult to achieve.

The EIP Problem

Except for Fiat-Shamir signatures, it is not known how to build cryptographic schemes solely on $\mathcal{MQ}$. Existing schemes rely, implicitly or explicitly, on at least a few other problems. The most prevalent is EIP, for *Extended Isomorphism of Polynomials* [DYC+08].

Extended Isomorphism of polynomials

Let $\mathcal{C}$ be a class of so-called quadratic *central maps* (from $\mathbb{F}^n$ to $\mathbb{F}^m$). Let $\mathsf{F} \in \mathcal{C}$ be a central map, and $\mathsf{S} : \mathbb{F}^n \to \mathbb{F}^n$ and $\mathsf{T} : \mathbb{F}^m \to \mathbb{F}^m$ be two affine maps.

Given $\mathsf{P} = \mathsf{S} \circ \mathsf{F} \circ \mathsf{T}$, find F.

Many multivariate schemes rely on EIP, in the sense that solving EIP implies breaking the scheme. For most of these schemes, the person who generated F knows an efficient way of computing preimages for it. Typically, this results in F having some special structure.

This asymmetry between F (easy to invert) and P (hard to invert) makes it tempting to build public-key cryptography based on EIP: the public key would be P and the private key would contain S, F and T.

4.2 Encryption

The first multivariate encryption scheme, C^*, was introduced in 1988 by Matsumoto and Imai [MI88]. Like most multivariate encryptiom schemes, it relies at a high level on the ideas described in the paragraph about the EIP problem.

Multivariate Encryption *à la* C^*

- The secret key consists of the maps S, F and T; all shall be easy to invert, and S, T are affine maps.
- The public key is the map $\mathrm{P} = \mathrm{S} \circ \mathrm{F} \circ \mathrm{T}$, which is expected hard to invert.
- The encryption of a message *msg* is:
$$ctxt = \mathrm{P}(msg).$$
- The decryption of a ciphertext *ctxt* is:
$$\mathrm{T}^{-1} \circ \mathrm{F}^{-1} \circ \mathrm{S}^{-1}(ctxt).$$

At a high level, this mechanism is not too different from what is done in code-based encryption schemes such as McEliece or Niederreiter:

- The central map F plays the same role as the generator matrix G, as it allows to solve an otherwise untractable problem;
- The affine maps S, T plays the same role as the permutation matrix P and invertible matrix S of McEliece, in the sense that they *hide* the structure of the central map.

While it may look simple to get public-key encryption from EIP using this blueprint, it has been notoriously hard to obtain secure schemes in practice. The original scheme by Matsumoto and Imai, as well as many subsequent schemes [TDTD13, PBD14, YS15], have been broken [Pat95, PPS17, CSV17], and only a handful of schemes remain unbroken. It is fair to say that building secure and efficient multivariate encryption schemes remains open.

4.3 Signatures

Multivariate signature schemes have been easier to obtain than encryption schemes. There exist schemes based both on the hash-then-sign paradigm, and the Fiat-Shamir paradigm.

4.3.1 Hash-then-Sign

The high-level construction for multivariate Hash-then-sign schemes may be seen as the “dual” of the construction for encryption schemes, and is described in the next block. We can see that the signing and verification procedures mirror the decryption and encryption procedures of Section 4.2, respectively. This is very similar to how the code-based CFS signature mirrors Niederreiter’s encryption scheme and, to a lesser extent, to how the RSA signature scheme mirrors the RSA encryption scheme.

Multivariate Hash-then-Sign

- The secret key are the maps S, F and T, which are easy to invert.
- The public key is the map $\mathrm{P} = \mathrm{S} \circ \mathrm{F} \circ \mathrm{T}$, expected hard to invert.
- The signature of a message *msg* is:
$$sig = \mathrm{T}^{-1} \circ \mathrm{F}^{-1} \circ \mathrm{S}^{-1} \circ H(msg).$$
- The verifier accepts a signature *sig* only if:
$$\mathrm{P}(sig) = H(msg).$$

The main difference among the schemes relying on this paradigm is the choice of the underlying field $\mathbb{F}$ and of the central map F. One of the more popular strategies is the *oil and vinegar* approach. This is for example the approach chosen by LUOV [BPSV19] and Rainbow [DCP+19]. Another popular approach relies on variants of *hidden field equations*, like GeMSS [CFM+17].

The public key P typically consists of m quadratic polynomials n variables over $\mathbb{F}$, so it is often large. On the other hand, each signature *sig* is essentially a vector in $\mathbb{F}^m$, and as a result signatures are often very small (a few hundred bytes).

Unfortunately, no multivariate hash-then-sign signature admits a security proof based on a standard assumption.

4.3.2 Fiat-Shamir with $\mathcal{MQ}$

The main idea of Fiat-Shamir signatures with $\mathcal{MQ}$ is that, for some publicly known map $\mathsf{F} : \mathbb{F}^m \to \mathbb{F}^n$, the secret key will be a vector $\mathsf{x} \in \mathbb{F}^m$, the public key will be $\mathsf{y} = \mathsf{F}(\mathsf{x}) \in \mathbb{F}^n$, and each signature will be a zero-knowledge proof that the signer knows x. An efficient way to do that was proposed by Sakumoto, Shirai and Hiwatari [SSH11]. Their key idea is to use the *polar* form of F, namely:

$$\mathsf{G}(\mathsf{x}_1, \mathsf{x}_2) = \mathsf{F}(\mathsf{x}_1 + \mathsf{x}_2) - \mathsf{F}(\mathsf{x}_1) - \mathsf{F}(\mathsf{x}_2).$$

A key property of G is that it is bilinear. [SSH11] leverage this property to split the private key in two, effectively enabling the construction of identification protocols based on the $\mathcal{MQ}$ problem. The authors proposed 3-pass (1 commitment, 1 challenge, 1 response) and 5-pass (2 commitments, 2 challenges, 1 response) identification protocols, but a conversion into a signature scheme was proposed only for the 3-pass protocol (via the Fiat-Shamir transform).

Another step in the direction of provably secure signatures from $\mathcal{MQ}$ was done by Chen, Hülsing, Rijneveld, Samardjiska and Schwabe [CHR+16]. As the signatures obtained from the 3-pass protocol from [SSH11] were too inefficient, [CHR+16] instead adapted the 5-pass identification protocol so that it could be converted into a signature scheme, once again via the Fiat-Shamir transform. The resulting scheme, MQDSS, is proven secure under the $\mathcal{MQ}$ assumption. The scheme SOFIA [CHR+18] uses the same ideas but with a proof against adversaries with stronger quantum capabilities.

For such schemes, the public and secret keys are usually very small since they are the vectors x and y, respectively. However, the underlying identification protocol only achieves a soundness of about $\frac{1}{2}$, thus the signing procedure requires to repeat this protocol several times. As a result the signatures are rather large (more than 30 kB).

Summary

Inception:	1988
Assumptions:	$\mathcal{MQ}$, EIP
Enc/KEM:	-
Signatures:	LUOV, MQDSS, Rainbow, GeMSS

5 Signatures from One-Way Functions

One-way functions are functions which are easy to compute, but hard to invert; for example, the hash functions SHA-2 and SHA-3 are assumed to be one-way functions.

The idea to build signatures from one-way functions was first proposed independently by Lamport [Lam79] and Merkle [Mer90] in 1979 (the work of Merkle was published 10 years later). Until recently, all the signatures based on one-way functions could be seen as loosely related to the hash-then-sign paradigm, but signatures relying on the Fiat-Shamir paradigm were recently proposed, and their philosophy is very different.

From a security viewpoint, these signatures are very appealing. Indeed, it is proven [Rom90] that signatures exist if and only if one-way functions exist. Consequently, in terms of underlying hypotheses, signatures based on one-way functions are the best one could possibly hope for. In addition, all existing constructions are provably secure.

Unfortunately, these perks come at the cost of efficiency; all existing schemes have slow signing procedures as well as large signatures. Some of these schemes achieve higher efficiency, but require in exchange to maintain an internal *state*; but this is not always possible, as some deployment contexts preclude this possibility. However, if one accepts these restrictions, these signatures provide strong security guarantees.

There also is one method for building an encryption scheme from minimal assumptions, however it is extremely inefficient [Mer78], and this seems to be inherent [BM09].

5.1 Hard Problems

Informally, a one-way function is a function easy to compute but hard to invert. One can define several hardness assumptions for one-way functions and related notions. The most common assumptions are preimage, second preimage and collision resistance. Preimage resistance of a function H essentially states that H is hard to invert for a specified output y. Second preimage resistance makes a slightly different statement.

The Preimage Problem

Let $H : X \to Y$ be a function, and $y \in Y$.
Find $x \in X$ such that $H(x) = y$.

The Second Preimage Problem

Let $H : X \to Y$ be a function, and $x_1 \in X$.
Find $x_2 \neq x_1$ such that $H(x_1) = H(x_2)$.

Collision resistance states it is hard to find two inputs that H maps to the same output.

The Collision Problem

Let $H : X \to Y$ be a function.
Find $x_1 \neq x_2$ such that $H(x_1) = H(x_2)$.

Preimage vs Second Preimage vs Collision

It is possible to build functions which are hard for any of these problems and easy for any other one (with the exception that collision resistance always implies second preimage resistance). However, these examples are quite contrived and mostly of theoretical interest.

The best classical attacks known for finding preimages on a generic function $H : \{0,1\}^* \to \{0,1\}^n$ require about 2^n operations, compared to $2^{n/2}$ operations for finding collisions. In practice, there exist hash functions for which collisions have been found, but no (second-) preimage attack has been mounted: this includes MD5 [dB94] and SHA-1 [SBK+17].

The situation of quantum attacks is less clear as it is a recent field. Grover's algorithm [Gro96] reduces the quantum cost of preimage search to $O(2^{n/2})$. Quantum speed-ups have been proposed for collision as well [BHT98, CNS17], their practical applicability is still being discussed.

5.2 Hash-based Signatures

Hash-based signatures are based on the observation that a hash function H allows to commit to a secret key, while hiding it. In this perspective, the public key will be a commitment of the private key.

Signing a message typically consists in revealing partial information so that the verifier can recompute the commitment and check it against the public key. A peculiar (but useful) property of hash-based signatures in general is that one can recover the public key from a valid signature and the associated message.

One-Time Signatures

One-Time Signature (Toy Example)

- The private key is a bitstring $sk = (sk_1, sk_2)$.
- The public key is $pk = (H^M(sk_1), H^M(sk_2))$, H^M denoting the M-times iteration of H
- The signature of a message msg in $\{0, \ldots, M\}$ is:

$$(sig_1, sig_2) = (H^{msg}(sk_1), H^{M-msg}(sk_2)).$$

- The verifier accepts the signature if and only if:

$$(H^{M-msg}(sig_1), H^{msg}(sig_2)) = pk.$$

The scheme described in the "One-Time Signature (Toy Example)" box puts into practice the high-level ideas described at the beginning ot this section.

Without knowing sk, the only way to forge a signature for msg is to invert H. Therefore, this scheme is secure under the preimage hardness of H. However, one can also show that given signatures for two different messages $msg_1 \neq msg_2$, one can compute a signature for any message $msg_1 < msg_3 < msg_2$. Therefore, this scheme is secure if it signs at most one message. These kind of schemes are called *one-time* signatures (or OTS), and this is obviously a huge limitation.

From One-Time to Few-Times: Merkle Trees

One-time signatures can be converted to few-time signatures (which means that a few messages can be signed) by using *Merkle trees* [Mer90], illustrated in Figure 4.

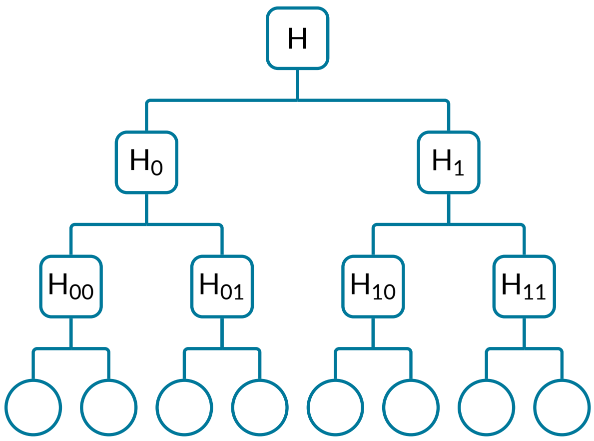


Figure 4: A Merkle tree

In a Merkle tree, each internal node (square nodes in Figure 4) is the hash of the concatenation of its children: this is illustrated by the edges linking these nodes to their children. For signatures, the leaves of a Merkle tree (circular nodes in Figure 4) are the public keys of OTS. The root of a Merkle tree (its top node) is a commitment of all the public keys (thus of the private keys as well).

Merkle Signature

- The private key is the set of all OTS private keys: $sk = \{sk_{000}, \ldots, sk_{111}\}$;
- The public key is the root of the Merkle tree: $pk = \mathrm{H}$;
- The signer uses a leaf OTS to sign *msg*, and sends the one-time signature along with nodes which allow to recover the root of the Merkle tree. For example, a valid signature may be:

$$sig = (sig_{000}, pk_{001}, \mathrm{H}_{01}, \mathrm{H}_1)\ ,$$

  where sig_{000} is a signature of *msg* using sk_{000} (the private key associated to pk_{000}).
- The verifier uses the key-recovery property of hash-based signatures to compute a public key from the one-time signature (here, he gets pk_{000} from sig_{000}), and recovers the top root of the Merkle tree from *sig*. For the example given above:

$$H(H(H(pk_{000} \| pk_{001}) \| \mathrm{H}_{01}) \| \mathrm{H}_1) = \mathrm{H} = pk.$$

  If it succeeds, the signature is accepted. Otherwise, it means that a part of the signature is incorrect, and it is rejected.

The Merkle tree in Figure 4 can sign up to 8 messages, which is better than one but is still far from perfect. Another caveat is that it still requires to keep track of the OTS keys used; indeed, once an OTS key is used, it can by definition not be reused. The "*bookkeeping*" that this method imposes on the signer is called *statefulness*, and it is an risky property that can be difficult to enforce, especially in distributed systems.

Going Stateless

Several other techniques allow to improve the efficiency and flexibility of hash-based signatures, and so we only briefly mention them here. Goldreich trees are a flexible variant of Merkle trees, which allow to relax the statefulness requirement to some extent. Stateless few-time signatures [RR02, updated version] also provide some extra flexibility. Hash-based signatures can be organized in two families:

- *Stateful* signatures still require to maintain a state. However, some of them attain reasonable signature sizes (less than 3 kB), such as LMS [LM95] and XMSS [HBG+18], which have recently been standardized by NIST [NIS20].
- *Stateless* signatures manage, by increasing parameters, to avoid the need for a state management. This comes at the cost of reduced efficiency; for example, the SPHINCS+ scheme yields signatures of about 30 kB.

Both families of schemes share several common points. For example, the public key is very small (around 64 bytes), because it is a single hash. Also, the signing procedure entails a large number of hash computations, and is therefore rather slow. Finally, because of the tree structure, these schemes only support a limited number of hashes; however, this number can be made arbitrarily large by setting the parameters adequately (SPHINCS+ supports up to 2^{64} messages).

5.3 Zero-Knowledge Signatures

In 2017, Chase et al. [CDG+17] proposed a novel way of using one-way functions to yield post-quantum signature schemes. Their work relies on the Fiat-Shamir transform, but also on other ideas, such as *secret sharing* and *multiparty computation*.

First proposed by Shamir [Sha79] and Blakley [Bla79], secret sharing consists of splitting a secret into *shares*, in a way that the secret can only be retrieved by combining sufficiently many shares. For example, given a binary value $b \in \{0, 1\}$, if we split b as $b = b_1 \oplus b_2$, where b_1 is uniformly random, then b_1 or b_2 alone do not provide any information about b, but knowing both allows to recover b.

Multiparty computation (or MPC for short) [Yao82, Yao86, GMW87], proposes solutions to perform computation over shared data, while preserving the purpose of secret sharing (that is, many shares must be combined to recover the secret). Multiparty computation has found many applications, like distributed computation, protection against side-channel attacks, etc.

MPC-in-the-head [IKOS07] performs an MPC computation and reveals intermediate data (though not enough to reveal any secret) in a pseudorandom manner. For example, given y, if one wants to prove that they know x such that $H(x) = y$, they can perform an MPC-in-the-head computation of $H(x)$ with, say, 3 shares, and send a transcript. This will convince a verifier (up to probability $1/3$) that the prover knows x. At a high level, this is not too different from how the Fiat-Shamir transform renders an identification protocol non-interactive.

[CDG+17] builds a signature scheme based on this idea (and others). This approach is quite different from hash-based signatures; these view a one-way function H as a black box, whereas the internal description of H is quite relevant for [CDG+17]. As each MPC-in-the-head transcript convinces the verifier with probability $1/3$, it needs to be repeated several times (200 to 800 in practice), which results in a slow signing procedure and large signatures. The resulting scheme, however, is as secure as the underlying function H. Picnic [ZCD+17] is an application of this idea.

Summary

Inception:	1979
Assumptions:	Collision or (second) preimage resistance of one-way functions
Enc/KEM:	-
Signatures:	XMSS, SPHINCS+, Picnic

6 Isogeny-based Cryptography

Isogeny-based cryptography is the youngest of the subfields of post-quantum cryptography studied in this document, since it really started in 2006.

For key-exchange and encryption, the idea is mostly to *revisit* the classical elliptic curve Diffie-Hellman and El-Gamal schemes, except that instead of working with points of elliptic curves, the elliptic curves themselves become the objects which are manipulated, and this is done through the use of *isogenies*, which are a class of maps between elliptic curves.

Signature schemes based on the Fiat-Shamir transform have been proposed recently.

For both key-establishment and signatures, this transposition has not been straightforward; the schemes are often not as "natural" as their elliptic curve counterparts, and efficiency issues are still being addressed at this time. While they are currently slow, these schemes offer excellent communication costs compared to schemes of other families.

This is a very recent field, so the security estimates, parameters and efficiency of schemes are likely to evolve.

6.1 Hard Problems

We recall that an elliptic curve is the set of points (x, y) that, for fixed (a, b), verify:

$$y^2 = x^3 + ax + b,$$

with a, b, x, y belonging to (the algebraic closure of) a finite field. Isogeny-based cryptography can work on two classes of elliptic curves, *ordinary* or *supersingular*. The second class currently seems to provide the best efficiency/security trade-off, and almost all schemes use it.

The Isogeny Problem

For the purpose of this document, it is sufficient to remember that isogenies are a specific class of maps between elliptic curves, and that isogenous curves are curves which are connected by an isogeny.

The Isogeny Problem

Given two supersingular isogenous curves E_1, E_2, compute an isogeny:

$$\varphi : E_1 \to E_2.$$

At a high level, we can see that this is similar to the discrete logarithm problem: instead of looking for a such that $g^a = h$, we look for an isogeny φ mapping E_1 to E_2.

6.2 Key-exchange

The idea of using isogenies to replicate the Diffie-Hellman protocol was first proposed by Couveignes [Cou06], and rediscovered by Rostovtsev and Stolbunov [RS06],

6.2.1 The Couveignes-Rostovtsev-Stolbunov scheme (CRS)

We recall that classical Diffie-Hellman starts with a public element g. Alice sends g^a, Bob sends g^b and at the end of the key-exchange they both have a shared secret g^{ab}. The Couveignes-Rostovtsev-Stolbunov scheme (or CRS) can be seen as a generalization of this idea, and is described in the next diagram, with the public element E at the top left, and the shared secret $E/\langle P, Q\rangle$ at the bottom right.

$$\begin{tikzcd} E \arrow[r, "\phi"] \arrow[d, "\psi"'] & E_{\phi} \arrow[d, "\psi"] \\ E_{\psi} \arrow[r, "\phi"] & E_{\psi,\phi} \end{tikzcd}$$

While classical (elliptic-curve) Diffie-Hellman takes g to be an element of an elliptic curve E, the CRS protocol subsitutes g with the elliptic curve E, and the action of exponentiating a point is replaced by an isogeny mapping E to another elliptic curve. Both Alice and Bob keep their isogenies (ϕ and ψ) secret, but at the end of the protocol they both share a known secret $E_{\psi,\phi}$.

Later, [CJS14] proposed quantum algorithms for computing isogenies in the CRS setting. To account for these algorithms, CRS required a growth in parameters which made it impractical at the time.

6.2.2 SIDH

Jao and de Feo [JD11] proposed an analog of the Diffie-Hellman key exchange over supersingular curves.

$$\begin{tikzcd} E \arrow[r, "\phi_A"] \arrow[d, "\phi_B"'] & E/\langle H_A \rangle \arrow[d, "{\phi_B, \{\phi_A(P_B), \phi_A(Q_B)\}}"] \\ E/\langle H_B \rangle \arrow[r, "{\phi_A, \{\phi_B(P_A), \phi_B(Q_A)\}}"] & E/\langle H_A, H_B \rangle \end{tikzcd}$$

This paradigm of key-exchange is generally called *Supersingular Isogeny Diffie-Hellman*, or SIDH. From a security perspective, this variant is somewhat incomparable to the ordinary case proposal from [Cou06, RS06]. On one hand, the [CJS14] algorithm does not apply in this case. On the other hand, the adversary has access to some additional information compared to [Cou06, RS06]. Therefore, the security of this scheme depends on a different problem: the SIDH problem.

The SIDH Problem

Given two supersingular isogenous curves E_1 and E_2, compute an isogeny

$$\varphi : E_1 \to E_2,$$

knowing $\varphi(P)$ and $\varphi(Q)$ for some precise P, Q (we omit the details here).

While no attack has exploited this additional information yet (except active attacks), it remains to be studied whether this introduces a weakness.

From a practical perspective, being able to work on supersingular curves allows much smaller parameters, which results in more compact and more efficient schemes. The KEM proposal SIKE essentially relies on SIDH combined with the HHK transform [HHK17].

6.2.3 Non-interactive key-exchange: CSIDH and revisited CRS

A nice property which the original Diffie-Hellman protocol relies on is *commutativity*:

$$(g^a)^b = (g^b)^a = g^{ab}.$$

The main issues with SIDH (somewhat ad-hoc problem and need for an interactive protocol) stem from the non-commutativity of the underlying operation. Therefore, very recent works have tried enforcing a commutative operation.

The first work, by [DKS18], revisited the CRS scheme, which already possessed this commutativity property but was impractical. The work of [DKS18] proposed mainly algorithmic improvements, and yields key-exchange on *ordinary* curves. It is still very slow.

On the other hand, [CLM+18] showed that, by restricting to subsets of the elliptic curves and of the isogenies, one could effectively construct a commutative group action over *supersingular* curves. Combining this with the algorithmic improvements of [DKS18] allowed them to adapt the CRS scheme over supersingular curves. The resulting scheme is called CSIDH. It remains rather slow, but small public key sizes make it attractive.

As a direct consequence of the commutativity of their underlying group actions, these two schemes are non-interactive, support static keys and are directly protected against active attacks (they do not require a CCA transform). The first two properties are desirable for several real-life applications, and the third one makes the schemes faster and simpler to implement. A caveat to both schemes (besides efficiency) is that the best quantum attacks are subexponential. In particular, recent works [BS20, Pei20, CCJR20] seem to imply that CSIDH requires much larger parameters than initially expected.

6.3 Signatures

While the first identification schemes (which can often be converted in signatures) have been proposed as early as 2014 [FJP14], concrete signature schemes based on isogenies have only appeared recently. One can expect significant improvements of the schemes and cryptanalysis in the future.

6.3.1 Early schemes

In 2017, two papers [YAJ+17, GPS17] proposed signature schemes based on supersingular isogenies. The first scheme is in both papers, and follows from the identification scheme from [FJP14]. The second one, proposed only in [GPS17], relied on more recent algorithmic techniques by [KLPT14].

The idea of [GPS17] is that, given three elliptic curves E, E_{com}, E_{pk} and two isogenies $\phi_{sk} : E \to E_{pk}$ and $\phi_0 : E \to E_{com}$, one can use the techniques of [KLPT14] to compute an isogeny $\phi_1 : E_{com} \to E_{pk}$, which is summarized below.

$$\begin{tikzcd} E \arrow[r, "\phi_0"] \arrow[dr, "\phi_{sk}"'] & E_{com} \arrow[d, dashed, "\phi_1"] \\ & E_{pk} \end{tikzcd}$$

We now describe a simplified version of the signature scheme from [GPS17].

GPS Protocol

- The public key consists of two curves E and E_{pk};
- The private key is an isogeny $\phi_{sk} : E \to E_{pk}$. This is done by first choosing E and ϕ_{sk} randomly and then computing E_{pk};
- A round of the protocol is as follows:
  - **Commit:** Choose a random isogeny ϕ_0 and compute $E_{com} = \phi_0(E)$;
  - **Challenge:** The challenge c is 0 or 1.
  - **Response:** If $c = 0$, then $rsp = (E_{com}, \phi_0)$. Else, $rsp = (E_{com}, \phi_1)$, where $\phi_1 : E_{com} \to E_{pk}$ is computed using [KLPT14].

  Return *rsp*.
- The verifier accepts $rsp = (E_{com}, \varphi_c)$ if and only if either:
  - $c = 0$ and φ_0 sends E to E_{com}.
  - $c = 1$ and φ_1 sends E_{com} to E_{pk}.

  φ_c should of course be an isogeny.

At a high level, the underlying identification protocol resembles the well-known proof system of [GMW86] for proving knowledge of an isomorphism between graphs. Both protocols have a soundness of $\frac{1}{2}$, which informally means that an illegitimate prover may falsely convince a verifier that he knows the secret key, with a probability $\frac{1}{2}$.

To reach cryptographic levels of confidence, between 128 and 256 repetitions are needed. This yields rather large signatures.

6.3.2 SeaSign

Recently, de Feo and Galbraith [FG19] have taken advantage of the commutative group action introduced in [CLM+18] to propose a shorter signature scheme.

An interesting idea of [FG19] is to minimize soundness error by having a large number n of curves $E_{pk}^{(i)}$ composing the public key, then using Merkle trees to prevent an explosion of the public key size. Interestingly, the scheme also relies on techniques first developed for lattice-based cryptography: the identification protocol uses the aborting techniques from [Lyu09], and this protocol is converted into a secure signature scheme using recent work from [KLS18] originally targeting lattice-based schemes. The resulting scheme is called SeaSign. As for [GPS17], the signature consists of proving the knowledge of an isogeny between two curves.

6.3.3 CSI-FiSh

Even more recently, Beullens, Kleinjung and Vercauteren [BKV19] proposed several efficiency improvements to SeaSign. In particular, they compute an efficient representation of a class group (an algebraic structure) underlying much of the computations in SeaSign. This efficient representation, along with other tricks, enables us to substantially improve the efficiency of the scheme.

The resulting scheme is called CSI-FiSh and is extremely compact. When optimizing for signature size, the size can be as small as 263 bytes for 128 bits of classical security; when optimizing for combined public key and signature sizes, the total size can be as small as 1468 bytes. However, it is also currently very slow. Note that these numbers only apply for the parameter set CSI-FiSh-512, the security of which is under scrutiny [BS20, Pei20, CCJR20] and might call for updated parameters.

6.3.4 SQISign

Very recently, De Feo, Kohel, Leroux, Petit and Wesolowski [DKL+20] proposed a new declination of the [GPS17] scheme. A notable improvement compared to previous schemes is that the base protocol has a very small soundness error (compared to the $1/2$ in [GPS17] or $1/(n+1)$ in SeaSign). Consequently, one single round of the protocol is sufficient and the resulting signature scheme is therefore extremely compact.

$$\begin{tikzcd}
E \arrow[r, "\phi_{com}"] \arrow[d, "\phi_{sk}"'] & E_{com} \arrow[d, "\psi_{chal}"] \\
E_{pk} \arrow[r, "\phi_{rsp}"] & E_{chal}
\end{tikzcd}$$

The principle of SQISign is illustrated above, and explained below.

- The public key is comprised of two elliptic curves E and E_{pk}.
- The private key is an isogeny ϕ_{sk} sending E to E_{pk}.
- A signature simulates a commit-challenge-response identification protocol, where:
  - The commitment is E_{com}.
  - The challenge E_{chal} is computed by applying an isogeny ψ_{chal} to E_{com}.
  - The response is an isogeny ϕ_{rsp} sending E_{pk} to E_{chal}.

Summary

Inception:	2006
Hard Problems:	Isogeny Problem, SIDH Problem, CSIDH Problem
Enc/KEM:	SIKE, CSIDH
Signatures:	CSI-FiSh [BKV19], SQISign [DKL+20]

References

[AAB+19] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loic Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Phillippe Gaborit, Gilles Zémor, Alain Couvreur, and Adrien Hauteville. RQC. Technical report, National Institute of Standards and Technology, 2019. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions.

[ABD+19] Nicolas Aragon, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Adrien Hauteville, Olivier Ruatta, Jean-Pierre Tillich, Gilles Zémor, Carlos Aguilar Melchor, Slim Bettaieb, Loic Bidoux, Magali Bardet, and Ayoub Otmani. ROLLO. Technical report, National Institute of Standards and Technology, 2019. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions.

[ABG+19] Nicolas Aragon, Olivier Blazy, Philippe Gaborit, Adrien Hauteville, and Gilles Zémor. Durandal: A rank metric based signature scheme. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 728–758. Springer, Heidelberg, May 2019.

[AES01] Advanced Encryption Standard (AES). National Institute of Standards and Technology (NIST), FIPS PUB 197, U.S. Department of Commerce, November 2001.

[AGH+17] Nicolas Aragon, Phillipe Gaborit, Adrien Hautevillle, Olivier Ruatta, and Gilles Zémor. RankSign. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

[BGG+14] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 533–556. Springer, Heidelberg, May 2014.

[BHT98] Gilles Brassard, Peter Høyer, and Alain Tapp. Quantum cryptanalysis of hash and claw-free functions. In Claudio L. Lucchesi and Arnaldo V. Moura, editors, *LATIN 1998*, volume 1380 of *LNCS*, pages 163–169. Springer, Heidelberg, April 1998.

[BKV19] Ward Beullens, Thorsten Kleinjung, and Frederik Vercauteren. CSI-FiSh: Efficient isogeny based signatures through class group computations. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 227–247. Springer, Heidelberg, December 2019.

[Bla79] G. R. Blakley. Safeguarding cryptographic keys. *Proceedings of AFIPS 1979 National Computer Conference*, 48:313–317, 1979.

[BM09] Boaz Barak and Mohammad Mahmoody-Ghidary. Merkle puzzles are optimal - an $O(n^2)$-query attack on any key exchange from a random oracle. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 374–390. Springer, Heidelberg, August 2009.

[Boy13] Xavier Boyen. Attribute-based functional encryption on lattices. In Amit Sahai, editor, *TCC 2013*, volume 7785 of *LNCS*, pages 122–142. Springer, Heidelberg, March 2013.

[BPSV19] Ward Beullens, Bart Preneel, Alan Szepieniec, and Frederik Vercauteren. LUOV. Technical report, National Institute of Standards and Technology, 2019. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions.

[BR95] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *EUROCRYPT'94*, volume 950 of *LNCS*, pages 92–111. Springer, Heidelberg, May 1995.

[BR98] Mihir Bellare and Phillip Rogaway. Pss: Provably secure encoding method for digital signatures, 1998.

[BS20] Xavier Bonnetain and André Schrottenloher. Quantum security analysis of CSIDH. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 493–522. Springer, Heidelberg, May 2020.

[CCJR20] Jorge Chávez-Saab, Jesús-Javier Chi-Domínguez, Samuel Jaques, and Francisco Rodríguez-Henríquez. The sqale of csidh: Square-root vélu quantum-resistant isogeny action with low exponents. Cryptology ePrint Archive, Report 2020/1520, 2020. https://eprint.iacr.org/2020/1520.

[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.

[CFM+17] A. Casanova, J.-C. Faugère, G. Macario-Rat, J. Patarin, L. Perret, and J. Ryckeghem. GeMSS. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

[CFS01] Nicolas Courtois, Matthieu Finiasz, and Nicolas Sendrier. How to achieve a McEliece-based digital signature scheme. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 157–174. Springer, Heidelberg, December 2001.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 523–552. Springer, Heidelberg, May / June 2010.

[CHR+16] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. From 5-pass MQ-based identification to MQ-based signatures. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 135–165. Springer, Heidelberg, December 2016.

[CHR+18] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. SOFIA: $\mathcal{MQ}$-based signatures in the QROM. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part II*, volume 10770 of *LNCS*, pages 3–33. Springer, Heidelberg, March 2018.

[CJS14] Andrew M. Childs, David Jao, and Vladimir Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. *J. Mathematical Cryptology*, 8(1):1–29, 2014.

[CKMS16] Sanjit Chatterjee, Neal Koblitz, Alfred Menezes, and Palash Sarkar. Another look at tightness II: Practical issues in cryptography. Cryptology ePrint Archive, Report 2016/360, 2016. `http://eprint.iacr.org/2016/360`.

[CLM+18] Wouter Castryck, Tanja Lange, Chloe Martindale, Lorenz Panny, and Joost Renes. CSIDH: An efficient post-quantum commutative group action. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 395–427. Springer, Heidelberg, December 2018.

[CNS17] André Chailloux, María Naya-Plasencia, and André Schrottenloher. An efficient quantum collision search algorithm and implications on symmetric cryptography. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part II*, volume 10625 of *LNCS*, pages 211–240. Springer, Heidelberg, December 2017.

[Cou06] Jean-Marc Couveignes. Hard homogeneous spaces. Cryptology ePrint Archive, Report 2006/291, 2006. `http://eprint.iacr.org/2006/291`.

[CSV17] Daniel Cabarcas, Daniel Smith-Tone, and Javier A. Verbel. Key recovery attack for ZHFE. In Tanja Lange and Tsuyoshi Takagi, editors, *Post-Quantum Cryptography - 8th International Workshop, PQCrypto 2017*, pages 289–308. Springer, Heidelberg, 2017.

[dB94] Bert den Boer and Antoon Bosselaers. Collisions for the compressin function of MD5. In Tor Helleseth, editor, *EUROCRYPT'93*, volume 765 of *LNCS*, pages 293–304. Springer, Heidelberg, May 1994.

[DCP+19] Jintai Ding, Ming-Shing Chen, Albrecht Petzoldt, Dieter Schmidt, and Bo-Yin Yang. Rainbow. Technical report, National Institute of Standards and Technology, 2019. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions`.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[Din12] Jintai Ding. Cryptographic systems using pairing with errors, 2012.

[DKL+20] Luca De Feo, David Kohel, Antonin Leroux, Christophe Petit, and Benjamin Wesolowski. SQISign: Compact post-quantum signatures from quaternions and isogenies. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part I*, volume 12491 of *LNCS*, pages 64–93. Springer, Heidelberg, December 2020.

[DKRV17] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. SABER. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.

[DKS18] Luca De Feo, Jean Kieffer, and Benjamin Smith. Towards practical key exchange from ordinary isogeny graphs. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 365–394. Springer, Heidelberg, December 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.

[DN12] Léo Ducas and Phong Q. Nguyen. Learning a zonotope and more: Cryptanalysis of NTRUSign countermeasures. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 433–450. Springer, Heidelberg, December 2012.

[DST19] Thomas Debris-Alazard, Nicolas Sendrier, and Jean-Pierre Tillich. Wave: A new family of trapdoor one-way preimage sampleable functions based on codes. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 21–51. Springer, Heidelberg, December 2019.

[DXL12] Jintai Ding, Xiang Xie, and Xiaodong Lin. A simple provably secure key exchange scheme based on the learning with errors problem. Cryptology ePrint Archive, Report 2012/688, 2012. http://eprint.iacr.org/2012/688.

[DYC+08] Jintai Ding, Bo-Yin Yang, Chia-Hsin Owen Chen, Ming-Shing Chen, and Chen-Mou Cheng. New differential-algebraic attacks and reparametrization of Rainbow. In Steven M. Bellovin, Rosario Gennaro, Angelos D. Keromytis, and Moti Yung, editors, *ACNS 08*, volume 5037 of *LNCS*, pages 242–257. Springer, Heidelberg, June 2008.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.

[FG19] Luca De Feo and Steven D. Galbraith. Seasign: Compact isogeny signatures from class group actions. EUROCRYPT, 2019. https://eprint.iacr.org/2018/824.

[FJP14] Luca De Feo, David Jao, and Jérôme Plût. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. *Journal of Mathematical Cryptology*, 8(3):209 – 247, 01 Sep. 2014.

[FO99a] Eiichiro Fujisaki and Tatsuaki Okamoto. How to enhance the security of public-key encryption at minimum cost. In Hideki Imai and Yuliang Zheng, editors, *PKC'99*, volume 1560 of *LNCS*, pages 53–68. Springer, Heidelberg, March 1999.

[FO99b] 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.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987.

[GGH97] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In Burton S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 112–131. Springer, Heidelberg, August 1997.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, 1979.

[GJSS01] Craig Gentry, Jakob Jonsson, Jacques Stern, and Michael Szydlo. Cryptanalysis of the NTRU signature scheme (NSS) from Eurocrypt 2001. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 1–20. Springer, Heidelberg, December 2001.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption and how to play mental poker keeping secret all partial information. In *14th ACM STOC*, pages 365–377. ACM Press, May 1982.

[GMW86] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design (extended abstract). In *27th FOCS*, pages 174–187. IEEE Computer Society Press, October 1986.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

[GPS17] Steven D. Galbraith, Christophe Petit, and Javier Silva. Identification protocols and signature schemes based on supersingular isogeny problems. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 3–33. Springer, Heidelberg, December 2017.

[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.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *28th ACM STOC*, pages 212–219. ACM Press, May 1996.

[GS02] Craig Gentry and Michael Szydlo. Cryptanalysis of the revised NTRU signature scheme. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 299–320. Springer, Heidelberg, April / May 2002.

[HBG+18] Andreas Huelsing, Denis Butin, Stefan-Lukas Gazdag, Joost Rijneveld, and Aziz Mohaisen. XMSS: eXtended Merkle Signature Scheme. RFC 8391, May 2018.

[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.

[HMA08] The keyed-hash message authentication code (hmac). National Institute of Standards and Technology (NIST), FIPS PUB 198-1, U.S. Department of Commerce, 2008.

[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.

[HPS01] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NSS: An NTRU lattice-based signature scheme. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 211–228. Springer, Heidelberg, May 2001.

[IKOS07] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *39th ACM STOC*, pages 21–30. ACM Press, June 2007.

[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.

[KLPT14] David Kohel, Kristin Lauter, Christophe Petit, and Jean-Pierre Tignol. On the quaternion ℓ-isogeny path problem. Cryptology ePrint Archive, Report 2014/505, 2014. http://eprint.iacr.org/2014/505.

[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.

[Lam79] Leslie Lamport. Constructing digital signatures from a one-way function. Technical Report SRI-CSL-98, SRI International Computer Science Laboratory, October 1979.

[LDK+19] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2019. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions.

[LM95] Frank T. Leighton and Silvio Micali. Large provably fast and secure digital signature schemes based on secure hash functions, 1995. Patent expired.

[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.

[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.

[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.

[McE78] Robert J. McEliece. A public-key cryptosystem based on algebraic coding theory. *JPL DSN Progress Report*, 44, 05 1978.

[Mer78] Ralph C. Merkle. Secure communications over insecure channels. *Commun. ACM*, 21(4):294–299, 1978.

[Mer90] Ralph C. Merkle. A certified digital signature. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 218–238. Springer, Heidelberg, August 1990.

[MI88] Tsutomu Matsumoto and Hideki Imai. Public quadratic polynominal-tuples for efficient signature-verification and message-encryption. In C. G. Günther, editor, *EUROCRYPT'88*, volume 330 of *LNCS*, pages 419–453. Springer, Heidelberg, May 1988.

[NAB+17] Michael Naehrig, Erdem Alkim, Joppe Bos, Léo Ducas, Karen Easterbrook, Brian LaMacchia, Patrick Longa, Ilya Mironov, Valeria Nikolaenko, Christopher Peikert, Ananth Raghunathan, and Douglas Stebila. FrodoKEM. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

[NIE86] H. NIEDERREITER. Knapsack-type cryptosystems and algebraic coding theory. *Prob. Control and Inf. Theory*, 15(2):159–166, 1986.

[NIS20] NIST. Recommendation for stateful hash-based signature schemes, 2020. https://doi.org/10.6028/NIST.SP.800-208.

[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.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 223–238. Springer, Heidelberg, May 1999.

[Pat95] Jacques Patarin. Cryptanalysis of the Matsumoto and Imai public key scheme of eurocrypt'88. In Don Coppersmith, editor, *CRYPTO'95*, volume 963 of *LNCS*, pages 248–261. Springer, Heidelberg, August 1995.

[PBD14] Jaiberth Porras, John Baena, and Jintai Ding. ZHFE, a new multivariate public key encryption scheme. In Michele Mosca, editor, *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014*, pages 229–245. Springer, Heidelberg, October 2014.

[Pei14] Chris Peikert. Lattice cryptography for the internet. In Michele Mosca, editor, *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014*, pages 197–219. Springer, Heidelberg, October 2014.

[Pei20] Chris Peikert. He gives C-sieves on the CSIDH. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 463–492. Springer, Heidelberg, May 2020.

[PPS17] Ray A. Perlner, Albrecht Petzoldt, and Daniel Smith-Tone. Total break of the SRP encryption scheme. In Carlisle Adams and Jan Camenisch, editors, *SAC 2017*, volume 10719 of *LNCS*, pages 355–373. Springer, Heidelberg, August 2017.

[Rab79] Michael O. Rabin. Digital signatures and public key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, Massachusetts Institute of Technology, January 1979.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *22nd ACM STOC*, pages 387–394. ACM Press, May 1990.

[RR02] Leonid Reyzin and Natan Reyzin. Better than BiBa: Short one-time signatures with fast signing and verifying. In Lynn Margaret Batten and Jennifer Seberry, editors, *ACISP 02*, volume 2384 of *LNCS*, pages 144–153. Springer, Heidelberg, July 2002.

[RS06] Alexander Rostovtsev and Anton Stolbunov. Public-Key Cryptosystem Based On Isogenies. Cryptology ePrint Archive, Report 2006/145, 2006. `http://eprint.iacr.org/2006/145`.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, 1978.

[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`.

[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.

[Sch90] Claus-Peter Schnorr. Efficient identification and signatures for smart cards (abstract) (rump session). In Jean-Jacques Quisquater and Joos Vandewalle, editors, *EUROCRYPT'89*, volume 434 of *LNCS*, pages 688–689. Springer, Heidelberg, April 1990.

[Sha79] Adi Shamir. How to share a secret. *Communications of the Association for Computing Machinery*, 22(11):612–613, November 1979.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th FOCS*, pages 124–134. IEEE Computer Society Press, November 1994.

[SSH11] Koichi Sakumoto, Taizo Shirai, and Harunaga Hiwatari. Public-key identification schemes based on multivariate quadratic polynomials. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 706–723. Springer, Heidelberg, August 2011.

[Ste94] Jacques Stern. A new identification scheme based on syndrome decoding. In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 13–21. Springer, Heidelberg, August 1994.

[Ste96] Jacques Stern. A new paradigm for public key identification. *IEEE Trans. Inf. Theory*, 42(6):1757–1768, 1996.

[TDTD13] Chengdong Tao, Adama Diene, Shaohua Tang, and Jintai Ding. Simple matrix scheme for encryption. In Philippe Gaborit, editor, *Post-Quantum Cryptography - 5th International Workshop, PQCrypto 2013*, pages 231–242. Springer, Heidelberg, June 2013.

[Vér96] Pascal Véron. Improved identification schemes based on error-correcting codes. *Appl. Algebra Eng. Commun. Comput.*, 8(1):57–69, 1996.

[Wil84] Hugh C. Williams. Some public key crypto-functions as intractable as factorization. In G. R. Blakley and David Chaum, editors, *CRYPTO'84*, volume 196 of *LNCS*, pages 66–70. Springer, Heidelberg, August 1984.

[YAJ+17] Youngho Yoo, Reza Azarderakhsh, Amir Jalali, David Jao, and Vladimir Soukharev. A post-quantum digital signature scheme based on supersingular isogenies. In Aggelos Kiayias, editor, *FC 2017*, volume 10322 of *LNCS*, pages 163–181. Springer, Heidelberg, April 2017.

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd FOCS*, pages 160–164. IEEE Computer Society Press, November 1982.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986.

[YS15] Takanori Yasuda and Kouichi Sakurai. A multivariate encryption scheme with Rainbow. In Sihan Qing, Eiji Okamoto, Kwangjo Kim, and Dongmei Liu, editors, *ICICS 15*, volume 9543 of *LNCS*, pages 236–251. Springer, Heidelberg, December 2015.

[ZCD+17] Greg Zaverucha, Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, and Daniel Slamanig. Picnic. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.