Post-Quantum Cryptography: State of the Art

Johannes A. Buchmann, Denis Butin, Florian Göpfert, Albrecht Petzoldt

Technische Universität Darmstadt, Fachbereich Informatik,
Hochschulstraße 10, 64289 Darmstadt, Germany

Abstract. Public-key cryptography is indispensable for cyber security. However, as a result of Peter Shor shows, the public-key schemes that are being used today will become insecure once quantum computers reach maturity. This paper gives an overview of the alternative public-key schemes that have the capability to resist quantum computer attacks and compares them.

Keywords: Public-Key Cryptography, Quantum Computing, Post-Quantum Cryptography

1 Introduction

Since its invention in the late 1970s, public-key cryptography has become a major enabler of cyber security. For example, the security of the TLS protocol that protects billions of Internet connections daily relies on public-key encryption and digital signatures. Today, mostly the RSA schemes are used. In addition, schemes based on elliptic curves are becoming more and more popular. For example, elliptic-curve based digital signatures are used in the German electronic ID card and in the German electronic passport. The security of these schemes relies on the hardness of the *integer factorization problem* and the *discrete logarithm problem*. In the integer factorization problem, the prime factors of a given positive integer have to be found. The discrete logarithm problem refers to finding the exponent x when two elements g and h of a finite group G are given where $h = g^x$. In elliptic curve schemes, this group is the group of points of an elliptic curve over a finite field.

In 1994, Peter Shor [68] discovered polynomial time quantum computer algorithms that solve the integer factorization problem and the discrete logarithm problem in the groups relevant for public-key cryptography. Therefore, all public-key cryptosystems that are currently used in practice will become insecure once sufficiently large quantum computers can be built. As can be seen from [73], there is considerable technological progress in designing quantum computers. As public-key cryptography is indispensable for cyber security, alternative public-key schemes that resist quantum computer attacks have to be developed. Such schemes are referred to as *post-quantum schemes*. This paper gives an overview of the current state of the art regarding post-quantum public-key cryptography.

The paper starts with a short introduction into public-key cryptography in Section 2. This section focuses on explaining the RSA public-key encryption and signature schemes as they are currently very much used in practice. In particular, this section shows how RSA relies on the integer factorization problem. Section 3 describes the relevance of public-key cryptography for securing IT applications in more detail. As a consequence, it becomes clear that today's cyber security relies on the hardness of integer factorization and computing discrete logarithms. Section 4 discusses the current knowledge regarding the hardness of integer factorization and its development, both for attackers with access to quantum computers, and ones without. This section also briefly discusses the hardness of computing discrete logarithms. It becomes clear that there is a need for the development of alternative public-key cryptosystems that resist quantum computer attacks. Once the need for post-quantum public-key cryptography is established, the question arises what it means for a public-key scheme to be secure in a post-quantum world. This question is answered in Section 5. Section 6 gives an overview of the algorithmic problems on which the security of current post-quantum proposals is based. Section 7 describes such schemes. In Section 8 the different proposals for post-quantum schemes are compared and open problems are presented.

2 Public-key cryptography

In 1976, Witfield Diffie and Martin Helman published their seminal paper "New Directions in Cryptography" [18]. In this paper they write "Widening applications of teleprocessing have given rise to a need for new types of cryptographic systems which minimize the need for secure key distribution channels and supply the equivalent of a written signature." This was the start of public-key cryptography, at least of its discussion in public. Before, in 1970, the concept of "non-secret encryption" had already been developed at the Government Communications Headquarters (GCHQ) in Great Britain by James Henry Ellis. In their paper, Diffie and Helman presented a way of exchanging keys over an insecure channel and stated: "We propose some techniques for developing public-key cryptosystems, but the problem is still largely open." A satisfactory solution was given by Rivest, Shamir, and Adleman in their work "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems" [64] where they propose the RSA public-key encryption and signature scheme. They received the Turing Award 2002 "for their ingenious contribution to making public-key cryptography useful in practice".

The idea of Rivest, Shamir, and Adleman is as follows. Suppose that we are given a finite group G. We assume that this group is multiplicatively written. We also assume that elements in the group can be selected (randomly), multiplied, and tested for equality without the knowledge of the group order $|G|$.

RSA relies on the fact that anyone can efficiently raise elements to eth powers in G where e is a positive integer that is coprime to $|G|$ without knowing $|G|$ but that extracting eth roots in G requires the knowledge of $|G|$. In fact, our assumptions regarding G imply that given a group element g anyone can compute $h = g^e$ using fast exponentiation (see [12]). However, the only generic method to extract the eth root g of h requires the knowledge of $|G|$. A positive integer d is calculated with

$$de \equiv 1 \bmod |G|. \tag{1}$$

Then

$$h^d = g^{de} = g^{1+k|G|} = g \cdot (g^k)^{|G|} = g \tag{2}$$

since group elements raised to the $|G|$th power yield 1 by Lagrange's theorem (see [12]).

In the situation described in the previous paragraph, public-key cryptography can be realized. For key generation, the group G and the exponent e are selected and the exponent d is calculated by solving the congruence (1) using the secret group order $|G|$. This can be done by means of the extended Euclidean algorithm (see [12]).

For public-key encryption, the plaintexts are the group elements. Encrypting a plaintext $g \in G$ means raising it to the eth power. So the ciphertext is $h = g^e$. This ciphertext can be decrypted by extracting the eth root as explained in [12]: $g = h^d$.

The corresponding digital signature scheme uses a cryptographic hash function $H : \{0,1\}^* \to G$. The signature of a document $x \in \{0,1\}^*$ is $s = H(x)^d$. This signature is the eth root of $H(x)$. So verification requires checking whether $s^e = H(x)$ which can be done using the public key only.

The question remains which finite group has the desired property of allowing computations while the group order is unknown. In their RSA system Rivest, Shamir, and Adleman use the multiplicative group $G = (\mathbb{Z}/n\mathbb{Z})^*$ of integers modulo a composite integer $n = pq$ where p and q are large prime numbers. Anyone who knows the modulus n can compute in G. However, determining its order $|G| = (p-1)(q-1)$ requires factoring n which means finding the prime factors p and q. Hence, the security basis of the RSA system is the intractability of the integer factorization problem for sufficiently large prime factors p and q.

Subsequently, ElGamal [25] proposed cryptosystems whose security is based on the problem of computing discrete logarithms in certain finite groups. This problem refers to finding the exponent x when two elements g and $h = g^x$ of a finite group are given. ElGamal used the multiplicative group of finite fields. Later, Koblitz and Miller [34,52] suggested using the group of points of an elliptic curve over a finite field.

3 The relevance of public-key cryptography

There can be no doubt that public-key cryptography is one of the most important foundations of modern cyber security.

The Transport Layer Security (TLS) protocol requires digital signatures and also, in most cases, public-key encryption. This protocol protects the confidentiality and integrity of billions of Internet connections daily, for example in e-banking and e-commerce applications, and email traffic.

Digital signatures establish the authenticity of software downloads which are used all over the Internet. Examples are operating system updates, application software updates, in particular downloads of apps for smartphones, and malware detection software. They prevent the distribution of malicious code such as

```
Shell.Exec("rmdir /Q /S C:\Windows\System32")
```

instead of the genuine software. Note that the above fake software would destroy the operating system and imagine the impact of such a fake update's distribution to all users of an operating system. As the example of malware detection software shows, the possibility of software updates is crucial for Internet security as new malware is produced on a regular basis. But in turn, such updates are only useful if their authenticity can be verified.

Digital signatures protect the authenticity of electronic ID cards and passports. For example, the data on the RFID chip of the German electronic passport are digitally signed. These data include the data that are printed on the passport, the image of the passport holder, and his or her fingerprints.

In the future, the importance of public-key cryptography will continue to grow, for example in the context of secure car-to-car and air traffic communication.

4 The hardness of factoring and computing discrete logarithms

As explained in Section 2, the security of current public-key cryptography relies on the hardness of the integer factorization and the discrete logarithm problems in certain groups. In fact, with a few exceptions that use elliptic curve cryptography, most applications use the RSA schemes. This means that current cyber security relies on the hardness of factoring positive integers n which are the product of two large primes. Today, the RSA moduli are at least of length 1024 bits. In many cases, applications have switched to 2048 or even 4096 bit moduli. So the question arises whether factoring such integers is intractable and remains hard in the future.

Examining the factorization history of Fermat numbers yields a first assessment of the hardness of the integer factoring problem. These numbers were studied by Pierre de Fermat in the seventeenth century. For a positive integer n, the nth Fermat number is defined to be $F_n = 2^{2^n} + 1$. So we have $F_0 = 3$, $F_1 = 5$, $F_2 = 17$, $F_3 = 257$, $f_4 = 65537$. These numbers are all prime numbers. However, $F_5 = 4294967297$ is not. It is divisible by 641 as was discovered in 1732 by Euler. Subsequently, more Fermat numbers were factored as Table 1 shows.

Table 1 shows that the Fermat numbers double in length when the index is incremented by 1. It is interesting to see that the seventh Fermat number was only factored in 1970 by Brillhart and Morrison [54]. A computer program and an advanced algorithm were necessary to achieve this. Also, Table 1 shows that there is significant progress in factoring. This progress is due to advances in factoring algorithms and computer technology.

We briefly review algorithmic progress for the integer factoring problem. For a positive integer n and two positive real numbers u, v, with $0 \leq u \leq v$ we set

$$L_n[u, v] = e^{v (\log n)^u \log\log n^{1-u}}. \qquad (3)$$

n	$F_n = 2^{2^n} + 1$	
0	3	prime
1	5	prime
2	17	prime
3	257	prime
4	65,537	prime
5	4,294,967,297	$= 641 \cdot 6,700,417$
6	18,446,744,073,709,551,617	$= 59,649,589,127,497,217 \cdot 5,704,689,200,685,129,054,721$
7	$2^{128} + 1$ (39 digits)	$= p$ (17 digits) $\cdot\, q$ (22 digits)
8	$2^{256} + 1$ (78 digits)	$= p($ 16 digits) $\cdot\, q$ (62 digits)
9	$2^{512} + 1$ (155 digits)	$= p($ 49 digits) $\cdot\, q$ (99 digits)
10	$2^{1024} + 1$ (309 digits)	$= 45,592,577 \cdot 6,487,031,809 \cdot p($ 40 digits) $\cdot\, q$ (252 digits)
11	$2^{2048} + 1$ (617 digits)	$= 319,489 \cdot 974,849 \cdot p($ 21 digits) $\cdot\, q$ (22 digits) $\cdot\, r$ (564 digits)

Table 1. Factorization of Fermat Numbers (until 2014)

This function is used to quantify the running time of factoring algorithms. We note that

$$L_n[0, v] = (\log n)^v. \tag{4}$$

So factoring algorithms with running time $L_n[0, v]$ run in polynomial time. Also, we have

$$L_n[1, v] = (e^{\log n})^v. \tag{5}$$

This shows that factoring algorithms with running time $L_n[1, v]$ run in exponential time. In other words, the function $L_n[u, v]$ can be viewed as a linear interpolation of linear and exponential running time.

Running times $L_n[u, v]$ with $0 < u < 1$ are called *subexponential.* The quadratic sieve, which was invented in 1981 by Carl Pomerance [62], has subexponential running time $L_n[1/2, 1 + o(1)]$ where $o(1)$ stands for a function that converges to 0 as n goes to infinity. In fact, all advanced algorithms discovered until the late eighties are of complexity $L_n[1/2, v]$ for some v, that is, they are "in the middle" between polynomial and exponential.

In 1990 John Pollard invented the number field sieve [39] which was later shown by Buhler, Lenstra, and Pomerance [15] to be of complexity $L_n[1/3, \sqrt[3]{64/9}]$. This was a big step forward. The number field sieve is still the fastest known factoring algorithm.

This development shows that factoring sufficiently large RSA moduli is still intractable but that unexpected breakthroughs are always possible. Currently, the largest RSA modulus that has been factored has 768 bits. It required the equivalent of almost 2000 years of computing on a single core 2.2 GHz AMD Opteron processor.

The situation for discrete logarithms in multiplicative groups of finite fields is similar to the factoring situation. Subexponential algorithms have been discovered for such groups [32]. However, in the group of points of elliptic curves over finite fields, only exponential discrete logarithm algorithms are known. This is why the key sizes in these cryptosystems are considerably smaller than RSA keys and applications start using elliptic curve cryptography.

The above only refers to algorithms for "conventional computers". In the early 1980s, Yuri Manin and Richard Feynman [23] came up with the concept of quantum computers. Such computers use the quantum mechanical phenomena of superposition and entanglement to speed up computations. In 1994 Peter Shor [68] presented polynomial-time algorithms for factoring integers and computing discrete logarithms. This means that the public-key schemes from Section 2 will be insecure once sufficiently powerful quantum computers are available. There is considerable technological progress in quantum computing [66]. So the development of large quantum computers in the near future cannnot be excluded.

5 Post-quantum security

As explained in Section 3, public-key cryptography is indispensable for the security of present and future computing infrastructures. Also, as shown in Section 4, the security of the public-key cryptography schemes that are currently being used in practice is threatened by quantum computers. Therefore, it is necessary to come up with alternative schemes that resist quantum computer attacks. They are called *post-quantum schemes*. Such schemes are *post-quantum secure*. In this section we discuss what is meant by this.

To define security of a cryptographic scheme the following is required. Firstly, there must be a protection goal that the cryptographic scheme is supposed to achieve. For example, encryption schemes protect *confidentiality* and digital signature schemes provide *integrity*, *authenticity*, and *non-repudiation* (for details see [48]). Secondly, there must be an adversary model that describes the goals of a potential adversary and the capabilities and resources that the adversary can use. For example, in the *ciphertext-only security model* for encryption schemes, the adversary searches for plaintexts that correspond to given ciphertexts and can only see ciphertexts. In *the chosen cipertext model*, the adversary can encrypt plaintexts of her choice. Thirdly, the time period for which a cryptographic scheme is supposed to achieve its security goals must be known. For example, one-time passwords only need to be kept confidential until they have been used while conventional passwords must be protected until they expire. Post-quantum security refers to the resources of a potential adversary: he has access to a quantum computer.

We describe how the security of a cryptographic scheme S is established. An algorithmic problem P is selected whose hardness guarantees the security of S. No polynomial time algorithm for solving P must be known as in this case P would be considered easy to solve. In the case of RSA, P is the integer factoring problem. Once quantum computers reach maturity, integer factoring can no longer be used as the security basis of cryptographic schemes since polynomial-time algorithms for integer factorization will be available.

If P cannot be solved in polynomial time, an instance of S is selected that achieves the desired security level. Such an instance is determined by choosing the necessary parameters and keys. For example, the RSA encryption scheme is instantiated by choosing the RSA modulus and the RSA encryption exponent. Likewise, the underlying algorithmic problem can be instantiated. In the case of the integer factorization problem, an instance is determined by the number to be factored. Each instance I_S of S is associated with an instance I_P of P whose intractability guarantees the security of I_S in the chosen security model. In order for I_S to be secure for a sufficiently long time period, the instance I_P must remain intractable during this time period. So there are two tasks in this context. Firstly, connecting I_S to some I_P and secondly, determining the hardness of the instances of P. The first task is either addressed using a mathematical reduction proof or, if this is not possible, by applying heuristical arguments. In the case of RSA and the relevant security models, no reduction proof is known. The second task is to analyze the hardness of P. Such an analysis provides a lower bound for the computational resources required to solve a given instance of P. There are different models for measuring the resources. An example is *dollar days*, where x dollar days refers to the computational power that can be bought for x dollars being available for one day. For details see [38].

As the necessary technical details about quantum computers are still unknown, such a more detailed analysis of post-quantum security is not yet possible. This is why post-quantum security currently refers to a cryptographic scheme being associated in the above sense to a computational problem that is not solvable in polynomial time on a quantum computer. This includes the impossibility of solving this problem on a conventional computer.

6 Post-quantum problems

Trying to find algorithmic problems that provably resist quantum computer attacks seems hopeless. There is not even such a problem that resists classical attacks. Therefore, the only possible strategy is to identify algorithmic problems for which the resistance to quantum computer attacks is plausible. Such a plausibility is currently based on two arguments. The first argument is that attempts of the scientific community to find polynomial time quantum algorithms for these problems have failed since a long time. The second argument is the belief that NP-hard problems resist quantum attacks. Such a belief is suggested by certain complexity theoretic arguments (see [5]). This suggests constructing post-quantum schemes based on NP-hard problems. However, there is a problem with this approach. The security of the currently discussed candidates for post-quantum schemes is based on subproblems of NP-hard problems which themselves are not proven to be NP-hard. So it appears that the first argument must be used. The second argument may enhance the plausibility.

We now review the algorithmic hardness assumptions that are currently being used as the security basis of post-quantum public-key cryptography.

6.1 Cryptographic hash functions

The first security assumption is the existence of a secure *cryptographic hash function*. This is a function that maps bit strings of arbitrary length to bit strings of a fixed *hash length n* and satisfies certain security and efficiency requirements. An important efficiency requirement is that software implementations of the hash functions are able to hash long strings very efficiently. Another efficiency requirement is that the hash function lends itself to high performance hardware implementations. The most basic security requirement is *collision resistance*, which means that finding two strings with the same hash is intractable. Collision resistance implies other weaker security requirements such as *one-wayness*, *second-preimage resistance*, and *target collision resistance* which are explained in [65]. In fact, for a complexity-theoretic reduction proof as mentioned in Section 5, an individual secure and efficient hash function is insufficient. Instead, a family of hash functions is required whose elements are parametrized by a security parameter.

There is currently no cryptographic hash function or hash function family that is efficient and provably secure. Because of the efficiency requirement, the practical hash functions use construction principals that may be exploited by attackers. This has happened in the past for the cryptographic hash function MD5 [72]. Therefore, cryptographic hash functions are used that appear to be secure after a thorough analysis of the scientific community. This analysis takes into account attackers that can use algorithms and computing resources available today or in the foreseeable future, including quantum attackers. Currently, there are several such hash functions, for example SHA-3 [8] and RIPEMD [21]. Generalizing these constructions, secure hash function families are obtained.

Although the impact of quantum computers cannot yet be estimated, a lower bound on an important parameter of hash functions can be given for the pre-quantum and post-quantum world: the hash length. Let h be a cryptographic hash function with hash length n. Suppose that collision resistance is desired of h. The generic *birthday attack* (see [12]), which works for any hash function, finds a collision in time approximately $2^{n/2}$. Hence, if n is chosen too small, then h is not collision resistant. Today, $n \geq 256$ is required to prevent such an attack. In the quantum world, there is a generic attack [11] that finds a collision in time approximately $2^{n/3}$. Therefore, $n \geq 384$ is required for collision resistance.

If only second-preimage resistance is required of h, then the birthday attack and its quantum generalizations do not work. On classical computers, second preimages can only be found by exhaustive search in time approximately 2^n. Therefore, in the world of classical computers, the hash length must be at least 128. In the quantum world, Grover's algorithm [28] can be used to find a second preimage in time approximately $2^{n/2}$. This leads to a lower bound of 256 for n.

6.2 Short vectors in lattices

An important class of computational problems that serve as the basis of post-quantum algorithms are *lattice-problems*. We define lattices and present some important lattice problems. Let n, k be positive integers and let $k \leq n$. Let $b_1, \dots, b_k$

be linearly independent vectors in real n-space $\mathbb{R}^n$. Write $B = (b_1, \dots, b_k)$ for the matrix with column vectors b_i, $1 \leq i \leq k$. Then the lattice $L(B)$ is the set $\{\sum_{i=1}^{k} x_i b_i : x_i \in \mathbb{Z}, 1 \leq i \leq k\}$ of all integer linear combinations of the vectors b_i. The lattices in real n-space are exactly the $L(B)$ for some B as above. The sequence $(b_1, \dots, b_k)$ is called a *basis* of $L(B)$. For $k > 2$, there are infinitely many bases of $L(B)$ which are all of length k. The set of all bases of $L(B)$ is $\{BT : T \in GL(k, \mathbb{Z})\}$ where $GL(k, \mathbb{Z})$ denotes the set of all k by k matrices of determinant ±1 with integer entries. As all bases of $L(B)$ have the same length k, this number is called the *dimension* of $L(B)$.

The first lattice problem that is used in cryptography is the *shortest vector problem SVP*. In this problem, n, k, and a basis B of a lattice L in $\mathbb{R}^n$ of dimension k are given. The task is to find a shortest non-zero lattice vector, typically in the Euclidean norm. This problem is known to be NP-hard under random reduction [1].

A related problem is the *closest vector problem CVP*. In addition to the input of SVP, a target vector $t \in \mathbb{R}^n$ is given. The goal is to find a lattice vector $v \in L$ such that the distance between v and t is minimal, that is $||v - t|| = \min_{w \in L} ||w - t||$. This problem is NP-hard [51]. In fact, in cryptography, approximate versions of these problems are used. To state these problems a function α is required that maps positive integers to real numbers ≥ 1. Then αSVP searches for a non-zero lattice vector v whose length is at most $\alpha(k)\lambda(L)$ where $\lambda(L)$ is the length of a shortest non-zero vector in L. Likewise, αCVP tries to find a vector $v \in L$ such that $||v - t|| \leq \alpha(k) \min_{w \in L} ||w - t||$. Such relaxed problems are also known to be NP-hard under random reduction for certain choices of α. However, for the α used in cryptography such statements are not known.

Another important lattice problem is *lattice basis reduction*. Given n, k and B the goal is to find a basis B' of $L(B)$ with short basis vectors. There are several notions of reduction. For example, *LLL-reduction* [37] is polynomial time while *Korkine-Zolotaref-reduction* is NP-hard. The importance of lattice basis reduction comes from the fact that the most efficient algorithms for solving αSVP and αCVP use lattice basis reduction as a subroutine. Also, the hardness of αSVP and αCVP depend on the input basis being reduced or not.

6.3 Decoding over finite fields

Another class of hard algorithmic problems that is used as a basis of post-quantum public-key cryptography comes from coding theory. Let $k \leq n$ be positive integers and let $\mathbb{F}$ be a finite field. The *Hamming weight* $w(u)$ of a vector $u \in \mathbb{F}^n$ is the number of nonzero components of u. The *Hamming distance* between two vectors u and v in $\mathbb{F}^n$ is $w(u - v)$. An $[n, k]$ *linear code* $\mathcal{C}$ is a k-dimensional subspace of $\mathbb{F}^n$. Such a code $\mathcal{C}$ can be defined as $\mathcal{C} = \{uG : u \in \mathbb{F}^k\}$ with a *generator matrix* $G \in \mathbb{F}^{k \times n}$. The set of all generator matrices of $\mathcal{C}$ is $\{TG : T \in GL(k, \mathbb{F})\}$. The *general decoding problem (GDP)* is defined as follows.

Problem GDP: Given integers t, n, k with $t \ll k < n$, a generator matrix G of $\mathcal{C}$, and a target vector $v \in \mathbb{F}^n$, find a code word $c \in \mathcal{C}$ such that $w(c - v) \leq t$.

Typically, t is chosen less than half the minimum distance of code words in $\mathcal{C}$. Then c is uniquely determined by v. Solving the decoding problem is also referred to as *correcting the errors in* v. Error correction plays an important role in electronic communication and storage. The general decoding problem is known to be NP-complete (see [26]).

There are linear codes having generator matrices that enable efficient decoding. Examples for such code are *binary Goppa codes*. They are linear codes over the field $\mathbb{F}$ of 2^m elements where m is a positive integer. A binary Goppa code is defined by a *Goppa polynomial* g which is a polynomial with coefficients in $\mathbb{F}$. Denote its degree by t. If g is irreducible over $\mathbb{F}$, then the minimum distance of two code words is $2t + 1$. Using such a Goppa polynomial, the decoding problem for Goppa codes can be solved in polynomial time for errors of weight at most t. Goppa codes with irreducible Goppa polynomial are called irreducible.

Again, code-based cryptosystems do not rely on the decoding problem in its full generality. Instead, they use codes such as Goppa codes [6] for which a representation exists that allows for efficient decoding.

6.4 Solving multivariate quadratic equations over finite fields

The last type of problems that support the security of post-quantum public-key cryptography comes from algebraic geometry. Let $\mathbb{F}$ be a finite field. The problem of *solving systems of quadratic equations over* $\mathbb{F}$ is defined as follows.

Problem MQ: Given positive integers m and n and m quadratic polynomials $p_1, \ldots, p_m$ in the n variables $x_1, \ldots x_n$ with coefficients in $\mathbb{F}$, find field elements $X_1, \ldots, X_n$ such that $p_j(X_1, \ldots, X_n) = 0 \ \forall j \in \{1, \ldots, m\}$.

The problem MQ is proven to be NP-complete (for $m \approx n$) [26]. However, most multivariate schemes use only subclasses of MQ.

7 Post-quantum public-key schemes

7.1 Hash-based signatures

In the late 1970s, not only RSA but also the Merkle signature scheme (MSS) [49,50] was invented. In contrast to the RSA or ElGamal schemes, it only relies on the security of its underlying cryptographic hash function. RSA and ElGamal also use such hash functions. But as explained in Section 4, their security also relies on the hardness of number-theoretic problems. The idea of MSS is as follows. MSS generates many pairs consisting of a signature key and the corresponding verification key for the Lamport-Diffie one-time signature scheme [35]. Since one-time signatures partially reveal the signing key, each one-time key can only be used once. MSS uses a complete binary hash tree to reduce the validity of 2^H one-time verification keys (with H being the tree's height) to one MSS public key. The leaves of this tree are the hashes of the one-time verification keys. Any inner node is the hash of the concatenation of its two children. The root of the tree is the MSS public key. When a signature is generated, the signer selects a secret signing key that has not been used yet and creates the one-time signature. The MSS signature consists of the one-time signature, the verification key, and the *authentication path* for the corresponding one-time verification key. The authentication path contains the siblings of the nodes in the path from the leaf corresponding to the verification key in the signature to the root of the hash tree. The verifier proceeds as follows. She verifies the one-time signature using the verification key, both contained in the signature. She then uses the authentication path to construct the root of the hash tree from the hash of the verification key. This root is compared with the MSS public key. The signature is validated by verifying the one-time signature.

When MSS was first proposed, the scheme was much less efficient than RSA and ElGamal. Meanwhile, several research projects have improved the situation both in regard to security and efficiency. Currently, the most advanced hash-based signature scheme is XMSS [14]. It uses multiple Merkle trees, as well as a pseudorandom number generator yielding reduced signing key storage requirements. XMSS only requires a target collision-resistant hash function to be secure. Any such hash function yields a new instance of XMSS. More generally, it is shown in [14] that there is a secure instance of XMSS as long as there is any secure signature algorithm. This means that XMSS has minimal security requirements. It also forward secure, i.e. signatures generated before a key compromise remain valid. Instead of Lamport-Diffie one-time signatures, it uses a variant of the more efficient Winternitz scheme [13]. Furthermore, XMSS is very efficient in practice as its implementation on different platforms shows. For details see [14].

The fact that an IETF Internet-draft for hash-based signatures [46] exists demonstrates their readiness for practical application. For their practical use, a major challenge is to deal with the most important difference between hash-based and other signature schemes: statefulness. This refers to the fact that hash-based signatures rely on many one-time key pairs, making it necessary to keep track of key usage. At any time the signing device must know the state of the scheme: which of the one-time signature keys have been used and which have not. If several signing devices are used, it is necessary to synchronize them on a regular basis or to partition the set of signing keys into disjoint subsets: one per device. Key storage demands special care in the case of statefulness, since new attack vectors may surface. Another challenge is the issue of optimal parameter selection, which is not obvious. This problem has been partially addressed in a recent generalization of XMSS [31].

7.2 Code-based public-key cryptography

In this section we show how to construct public-key encryption and digital signature schemes based on coding theory. The public-key encryption schemes are very efficient except for the large key sizes. The digital signature schemes still require more research.

An important example of a code-based public-key encryption scheme is the McEliece scheme [45] invented in 1978. This scheme is still considered to be secure, even in a quantum world.

To generate a key pair for this scheme, one selects a generator matrix G for a binary $[n, k]$ code $\mathcal{C}$ such that G can be used to efficiently correct t errors for some t much less than the minimum distance of C. Also, a non-singular matrix $S \in \mathbb{F}^{k \times k}$ and a permutation matrix $P \in \mathbb{F}^{n \times n}$ are selected randomly with the uniform distribution. They are used to hide the special generator matrix G by computing the generator matrix $G' = SGP$ of the permuted code $\mathcal{C}'$. Then, the public key of the scheme is (G', t). The secret key consists of S, G and P.

To encrypt a message $m \in \mathbb{F}^k$, one chooses randomly a vector $z \in \mathbb{F}^n$ of weight t. The ciphertext of the message m is $c = mG' + z \in \mathbb{F}^n$.

To decrypt the cipertext c, one proceeds as follows. First, we observe that

$$x = cP^{-1} = (mG' + z)P^{-1} = mSG + zP^{-1}. \tag{6}$$

Since P is a permutation matrix and the weight of z is t, the weight of zP^{-1} is also t. Furthermore, mSG is a codeword in $\mathcal{C}$ since G is a generator matrix of $\mathcal{C}$. This shows that the distance of x from $\mathcal{C}$ is t. Hence, by our assumption, x can be decoded, the result being the codeword

$$y = mSG. \tag{7}$$

Next, m can be calculated by solving the linear system (7). The original McEliece scheme uses binary Goppa codes.

Encryption and decryption in the McEliece scheme can be performed very efficiently [9]. However, the keys are quite large. There are variants of McEliece which deal with the problem of large key sizes, for example the scheme of Sendrier et al. [53] which allows fast hardware implementations [29].

For the McEliece scheme to be applicable in practice, a semantically secure conversion is needed. The idea of Persichetti [56] is going into the right direction.

Code-based signatures still require more research. There is the scheme of Courtois, Finiasz, and Sendrier [36]. In this scheme signing is quite slow and public key sizes are very large. Also, no security reductions are known. There are also signature schemes that follow the Fiat-Shamir paradigm (see [12]) and are based on the Stern identification protocol [70], such as [47,2]. This scheme has security proofs which reduce hard coding problems to the security of the schemes. Still, there are several efficiency issues with these schemes, for example the signature length.

7.3 Lattice-based public-key cryptography

Lattice-based cryptography is similar to code-based cryptography. The similarity comes from the fact that the knowledge of an in some sense reduced lattice basis allows computing closest vectors while this problem becomes intractable when an unstructured basis is given. This suggests making the reduced basis of a lattice $L \subset \mathbb{R}^n$ for some n the secret key and some other unstructured basis the public key. Encryption would mean to select a lattice vector v as plaintext and to hide it by adding some small error vector e: $c = v + e$. Decryption would be done by solving the closest vector problem with input c. The closest vector is the plaintext v. Likewise, a message d would be signed using a hash function $h : \{0,1\}^* \to \mathbb{R}^n$. The signature of a message d would be the closest vector s to $h(d)$. Verification would be performed by checking that s is a lattice vector that is close to $h(d)$.

Unfortunately, this straightforward approach does not yield secure schemes. It requires substantial modifications. Recent examples of lattice based public-key encryption schemes are [40,69] and of lattice-based signature schemes are [22,3]. As a result, schemes are obtained that do not directly rely on the hardness of lattice problems. Instead, they rely on the *learning with errors problem (LWE)* [63] and the *shortest integer solution problem (SIS)* [1]. In turn, the hardness of these problems is based on the hardness of lattice problems.

Lattice-based public-key cryptography is very promising. On the one hand, the schemes appear to be very efficiently implementable. If schemes are selected that rely on hard problems in *ideal lattices* [42,43] then the required storage space and computing time is very limited, at least asymptotically. On the other hand, many lattice-based schemes have very strong security properties: they allow a worst-to-average case security reduction. This means that an instance of the scheme under consideration is secure as long as the worst case of a large class of lattice problems is intractable. Why is this important? For other schemes such as RSA, code-based, hash-based, and multivariate schemes it can only be shown that an instance of a scheme is secure as long as a related computational problem is hard. So in order to guarantee security it is necessary to select the instance of the scheme in such a way that the corresponding instance of the computational problem is hard to solve. For RSA, it is known how to select hard instances: the RSA modulus is constructed as the product of two big random prime numbers. However, for the other types of schemes, it is not so clear how to construct hard instances of the underlying problem. Worst-to-average circumvents the necessity to generate hard instances.

However, when worst-to-average case reduction is used, the lattice-based schemes loose their efficiency. In order to make them more efficient, reductions to random instances of the LWE or SIS problem can be used. This results in very efficient schemes [22]. Another alternative is the NTRU public-key encryption scheme [30] which has no security proof whatsoever but very good performance.

In addition to the advantages explained in the previous sections, lattice-based cryptography allows for the implementation of many advanced schemes, most notably fully homomorphic encryption [27]. This is not known for the other classes of public-key schemes mentioned in this paper.

7.4 Multivariate public-key cryptography

In this section we explain how public-key schemes based on the hardness of solving systems of nonlinear multivariate equation over finite fields work in principle.

Let $\mathbb{F}$ be a finite field and m, n be two positive integers. One chooses a *central map*, which is a quadratic map $\mathcal{Q} : \mathbb{F}^n \to \mathbb{F}^m$, $x = (x_1, \ldots, x_n) \mapsto (q_1(x), \ldots, q_m(x))$. The map $\mathcal{Q}$ must be easily invertible in the sense that it is easy to find a preimage for every image x under $\mathcal{Q}$. To hide the structure of this central map in the public key one composes it with two affine linear maps $\mathcal{S} : \mathbb{F}^n \to \mathbb{F}^n$ and $\mathcal{T} : \mathbb{F}^m \to \mathbb{F}^m$. The result is the quadratic map $\mathcal{P} = \mathcal{T} \circ \mathcal{Q} \circ \mathcal{S}$ which is the public key of the corresponding public-key schemes. $\mathcal{P}$ is supposed to look like a random system and therefore is assumed to be difficult to invert. The secret key of the scheme consists of $\mathcal{Q}, \mathcal{S}$ and $\mathcal{T}$ and therefore allows to invert the public key.

To sign a document d one uses a hash function $\mathcal{H} : \{0,1\}^* \to \mathbb{F}^m$ to compute a hash value $h = \mathcal{H}(d)$ of the message. To compute a signature of the document d, the signer computes recursively $x = \mathcal{T}^{-1}(h),\ y = \mathcal{Q}^{-1}(x)$ and $z = \mathcal{S}^{-1}(y)$. The signature of the document d is $z \in \mathbb{F}^n$. Here, $\mathcal{Q}^{-1}(y)$ means finding a preimage of y under the central map $\mathcal{Q}$.
To verify the authenticity of a signature $z \in \mathbb{F}^n$, the receiver of a message checks if $\mathcal{P}(z) = \mathcal{H}(d)$.

There exists a large variety of practical multivariate signature schemes. The best known of these are UOV [33], Rainbow [19], and pFlash [20]. Additionally, there exist multivariate signature schemes from the HFEv- family, which produce very short signatures (e.g. 120 bit). The most promising scheme in this direction is Gui [61]. Signing and verifying with all of these schemes is very fast, presumably much faster than RSA and ECC [16,10].
In the last years, there have been several attempts to decrease the key size of multivariate signature schemes [58,59]. However, despite of this work, the key sizes of multivariate signature schemes are still much larger than those of classical schemes such as RSA.

To construct a public-key encryption scheme on the upper principle the central map $\mathcal{Q}$ must be injective. In particular, we therefore need $m \geq n$. To encrypt a plaintext $x = (x_1, \ldots, x_n) \in \mathbb{F}^n$, one simply computes $c = \mathcal{P}(x) \in \mathbb{F}^m$. Since the owner of the secret key can invert the central map $\mathcal{Q}$ and knows the two affine linear maps $\mathcal{S}$ and $\mathcal{T}$, she can determine the plaintext by computing $x = \mathcal{S}^{-1} \circ \mathcal{Q}^{-1} \circ \mathcal{T}^{-1}(c)$.

The currently most promising multivariate encryption scheme is the SimpleMatrix (or ABC) encryption scheme [71], which allows very fast en- and decryption. However, decryption failures occur with non-negligible probability. Furthermore, the key sizes of this scheme are quite large.
A major problem of all multivariate public-key schemes is their security, which is somewhat unclear. Many of the proposed multivariate schemes have been broken

in the past (e.g. MI [44] and Oil and Vinegar [55]). The above mentioned schemes are all quite young (some less than 10 years), which means that they have not yet been subject to extensive cryptanalysis. Furthermore, there exist no security proofs for multivariate public-key schemes.

However, similarly to the case of code-based cryptography, there exists a provable secure multivariate identification scheme [67]. Via the Fiat-Shamir transform [24] it is possible to extend this scheme to provable secure (yet inefficient) multivariate (ring) signature schemes [60].

8 Conclusion

In this section we evaluate and compare the proposals for post-quantum public-key cryptography that are described in this paper.

Firstly, we compare the required hardness assumptions. From a structural point of view, the general decoding problem, the lattices problems, and the problem of solving multivariate systems of quadratic equations over finite fields are similar. These problems are NP-hard. However, the computational problems that support the security of the post-quantum schemes are in subclasses that are not known to be NP-hard. As for the integer factorization problem, it may happen that algorithms are discovered that solve the relevant problems in polynomial time. In this case the corresponding cryptographic systems can no longer be considered to be secure. Therefore, thorough research is required to establish the hardness of these problems. In fact, in order to enable the selection of secure parameters for required security levels, such research must determine quantitatively the resources necessary to solve a given instance of the computational problems.

In code-based security, solving the general decoding problem for Goppa codes can be considered to be hard. Detailed studies investigate this hardness in details (see [7,57]). In contrast, structured codes such as quasi-linear codes must be studied in more detail. Lattice-based cryptography uses a multitude of different computational hardness assumptions such as LWE, ring LWE, SIS, ring SIS, αSVP, etc. Therefore, more research is required to establish the hardness of the most relevant of these problems in detail even though there is much research on the general lattice problems such as SVP (e.g. [17,41,4]). In multivariate cryptography, new research is required whenever a new central map is introduced or a sub-problem that allows for smaller keys.

The problem of coming up with a secure hash function appears to be quite different from the problems discussed in the previous section. Firstly, the only relevant security parameter is the hash length while the other problems have many more parameters. Secondly, experience shows that it is not hard to come up with a secure alternative if a cryptographic hash function is broken. Typically, an easy way of enhancing the security of a given hash function is to increase the number of rounds in the insecure hash function at the expense of reducing its efficiency. But also new constructions are possible. Thirdly, assuming that there is a secure hash function is much more basic than the problems from the previous section. For example, all signature schemes require such hash functions if long documents are to be signed.

Next, we compare the post quantum schemes that have been presented in this paper. The most advanced scheme is the hash-based signature scheme XMSS [14]. Furthermore, it provides the strongest security guarantees as it can be shown that there is a secure instance of XMSS as long as there is any secure signature scheme. This is due to the flexibility of XMSS: any secure cryptographic hash function can be used to construct a secure instance of XMSS. These properties support the quantum-resistance of XMSS. If a cryptographic hash function turns out to be vulnerable to quantum attacks — which has never happened so far — an alternative quantum-resistant hash function can be used to make XMSS quantum-resistant again. XMSS is also very efficient and there is even a related standard draft [46]. There are no hash-based public-key encryption schemes.

The fact that a practical and secure post-quantum signature scheme exists is consequential. In order to prepare computing systems for the quantum computer era, a quantum-immune trust anchor is needed for potential updates. XMSS can serve as such a trust anchor. Now is the time to integrate XMSS into standard protocols such as TLS or S/MIME and to develop concepts to deal with its statefulness.

A good alternative for hash-based signature schemes are multivariate signature schemes such as Rainbow [19]. Multivariate schemes offer fast signature generation and verification and produce significantly shorter signatures than RSA. However, the key sizes of multivariate signature schemes are still relatively large. Furthermore, there are no security proofs for the efficient multivariate signature schemes.

As for post-quantum public-key encryption schemes, currently the code-based McEliece scheme appears to be the most reliable choice. McEliece and RSA were proposed roughly at the same time and remain secure since then although there are no formal security proofs for them. The drawback of the McEliece scheme are its large keys. It is therefore not applicable in all contexts, for example, when there are limited computing resources. The usefulness of code-based signature schemes is unclear as they are still much too inefficient.

From a research and development point of view, lattice-based cryptography is very promising. There are very interesting proposals for signature and public-key encryption schemes. In addition, in lattice-based cryptography there are several special-purpose schemes for example for fully homomorphic encryption, blind signatures, ring signatures, and group signatures. Such schemes admit the strongest security proofs: worst-to-average-case reductions. The ring variants promise high efficiency as their time and space requirements are quasi-linear in the security parameter. However, lattice-based schemes still require more research before becoming practical. The hardness of the underlying problems, in particular of the relevant ideal-lattice problems, requires more research. There is still a gap between security and efficiency. The efficient versions do not yet take advantage of the strong reduction proofs. In many cases, reduction proofs would profit from becoming tighter.

How far is post-quantum cryptography? There are many promising proposals some of which are rather close to becoming practical. In view of the importance of public-key cryptography explained in Section 3, a joint effort is necessary to make the promising proposals ready for practice. Such an effort provides quantitative predictions of the hardness of the relevant problems and tight security proofs leading to trustworthy parameters. It also provides optimized implementations and standards.

References

1. M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, STOC '96, pages 99–108, New York, NY, USA, 1996. ACM.
2. S. M. E. Y. Alaoui, P. Cayrel, and M. Meziani. Improved identity-based identification and signature schemes using quasi-dyadic goppa codes. In T. Kim, H. Adeli, R. J. Robles, and M. O. Balitanas, editors, *Information Security and Assurance - International Conference, ISA 2011, Brno, Czech Republic, August 15-17, 2011. Proceedings*, volume 200 of *Communications in Computer and Information Science*, pages 146–155. Springer, 2011.
3. S. Bai and S. Galbraith. An improved compression technique for signatures based on learning with errors. In J. Benaloh, editor, *Topics in Cryptology - CT-RSA 2014*, volume 8366 of *Lecture Notes in Computer Science*, pages 28–47. Springer International Publishing, 2014.
4. S. Bai and S. Galbraith. Lattice decoding attacks on binary LWE. In W. Susilo and Y. Mu, editors, *Information Security and Privacy*, volume 8544 of *Lecture Notes in Computer Science*, pages 322–337. Springer International Publishing, 2014.
5. C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani. Strengths and weaknesses of quantum computing. *SIAM J. Comput.*, 26(5):1510–1523, Oct. 1997.
6. D. Bernstein, J. Buchmann, and E. Dahmen, editors. *Post Quantum Cryptography.* Springer, 2008.
7. D. Bernstein, T. Lange, C. Peters, and P. Schwabe. Faster 2-regular information-set decoding. In Y. Chee, Z. Guo, S. Ling, F. Shao, Y. Tang, H. Wang, and C. Xing, editors, *Coding and Cryptology*, volume 6639 of *Lecture Notes in Computer Science*, pages 81–98. Springer Berlin Heidelberg, 2011.
8. G. Bertoni, J. Daemen, M. Peeters, and G. V. Assche. The Keccak reference, January 2011. `http://keccak.noekeon.org/`.
9. B. Biswas and N. Sendrier. McEliece cryptosystem implementation: Theory and practice. In *PQCrypto*, volume 5299 of *Lecture Notes in Computer Science*, pages 47–62, 2008.
10. A. Bogdanov, T. Eisenbarth, A. Rupp, and C. Wolf. Time-area optimized public-key engines: MQ-cryptosystems as replacement for elliptic curves? In E. Oswald and P. Rohatgi, editors, *Cryptographic Hardware and Embedded Systems - CHES 2008*, volume 5154 of *Lecture Notes in Computer Science*, pages 45–61. Springer Berlin Heidelberg, 2008.
11. G. Brassard, P. Hoyer, and A. Tapp. Quantum algorithm for the collision problem. *arXiv preprint quant-ph/9705002*, 1997.
12. J. Buchmann. *Introduction to Cryptography.* Springer, 2004.

13. J. Buchmann, E. Dahmen, S. Ereth, A. Hülsing, and M. Rückert. On the Security of the Winternitz One-Time Signature Scheme. In A. Nitaj and D. Pointcheval, editors, *AFRICACRYPT*, volume 6737 of *Lecture Notes in Computer Science*, pages 363–378. Springer, 2011.
14. J. Buchmann, E. Dahmen, and A. Hülsing. XMSS — A Practical Forward Secure Signature Scheme Based on Minimal Security Assumptions. In B.-Y. Yang, editor, *PQCrypto*, volume 7071 of *Lecture Notes in Computer Science*, pages 117–129. Springer, 2011.
15. J. P. Buhler, H. W. Lenstra Jr, and C. Pomerance. Factoring integers with the number field sieve. In *The development of the number field sieve*, pages 50–94. Springer, 1993.
16. A. I.-T. Chen, M.-S. Chen, T.-R. Chen, C.-M. Cheng, J. Ding, E. L.-H. Kuo, F. Y.-S. Lee, and B.-Y. Yang. SSE implementation of multivariate PKCs on modern x86 CPUs. In *CHES*, volume 5747 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 2009.
17. Y. Chen and P. Q. Nguyen. Bkz 2.0: Better lattice security estimates. In D. Lee and X. Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 1–20. Springer Berlin Heidelberg, 2011.
18. W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.
19. J. Ding and D. Schmidt. Rainbow, a new multivariable polynomial signature scheme. In *ACNS*, volume 3531 of *Lecture Notes in Computer Science*, pages 164–175. Springer, 2005.
20. J. Ding, B.-Y. Yang, V. Dubois, C.-M. Cheng, and O. Chen. Breaking the symmetry: a way to resist the new differential attack. http://eprint.iacr.org/2007/366, 2007.
21. H. Dobbertin, A. Bosselaers, and B. Preneel. RIPEMD-160: A strengthened version of RIPEMD. In D. Gollmann, editor, *Fast Software Encryption, Third International Workshop, Proceedings*, volume 1039 of *Lecture Notes in Computer Science*, pages 71–82. Springer, 1996.
22. L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky. Lattice signatures and bimodal gaussians. In R. Canetti and J. Garay, editors, *Advances in Cryptology - CRYPTO 2013*, volume 8042 of *Lecture Notes in Computer Science*, pages 40–56. Springer Berlin Heidelberg, 2013.
23. R. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6-7):467–488, 1982.
24. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.
25. T. E. Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18. Springer, 1984.
26. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.
27. C. Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 169–178, New York, NY, USA, 2009. ACM.
28. L. K. Grover. A Fast Quantum Mechanical Algorithm for Database Search. In G. L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 212–219. ACM, 1996.

29. S. Heyse, I. von Maurich, and T. Güneysu. Smaller keys for code-based cryptography: QC-MDPC McEliece implementations on embedded devices. In *CHES*, volume 8086 of *Lecture Notes in Computer Science*, pages 273–292. Springer, 2013.
30. J. Hoffstein, J. Pipher, and J. Silverman. Ntru: A ring-based public key cryptosystem. In J. Buhler, editor, *Algorithmic Number Theory*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer Berlin Heidelberg, 1998.
31. A. Hülsing, L. Rausch, and J. Buchmann. Optimal Parameters for XMSS^{MT}. In A. Cuzzocrea, C. Kittl, D. E. Simos, E. Weippl, and L. Xu, editors, *CD-ARES Workshops*, volume 8128 of *Lecture Notes in Computer Science*, pages 194–208. Springer, 2013.
32. A. Joux. A new index calculus algorithm with complexity l (1/4+ o (1)) in very small characteristic. *IACR Cryptology ePrint Archive*, 2013:95, 2013.
33. A. Kipnis, J. Patarin, and L. Goubin. Unbalanced oil and vinegar schemes. In *Proceedings of EUROCRYPT 1999*, volume 1592 of *Lecture Notes in Computer Science*, pages 206–222. Springer, 1999.
34. N. Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):pp. 203–209, 1987.
35. L. Lamport. Constructing Digital Signatures from a One Way Function. Technical report, SRI International Computer Science Laboratory, 1979. `http://research.microsoft.com/en-us/um/people/lamport/pubs/dig-sig.pdf`.
36. G. Landais and N. Sendrier. Implementing CFS. In *INDOCRYPT*, volume 7668 of *Lecture Notes in Computer Science*, pages 474–488. Springer, 2012.
37. A. Lenstra, J. Lenstra, H.W., and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.
38. A. K. Lenstra. Key lengths. Technical report, Wiley, 2006.
39. A. K. Lenstra, H. W. L. Jr., M. S. Manasse, and J. M. Pollard. The number field sieve. In H. Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 564–572. ACM, 1990.
40. R. Lindner and C. Peikert. Better key sizes (and attacks) for LWE-based encryption. In A. Kiayias, editor, *Topics in Cryptology - CT-RSA 2011*, volume 6558 of *Lecture Notes in Computer Science*, pages 319–339. Springer Berlin Heidelberg, 2011.
41. M. Liu and P. Nguyen. Solving bdd by enumeration: An update. In E. Dawson, editor, *Topics in Cryptology - CT-RSA 2013*, volume 7779 of *Lecture Notes in Computer Science*, pages 293–309. Springer Berlin Heidelberg, 2013.
42. V. Lyubashevsky and D. Micciancio. Generalized compact knapsacks are collision resistant. In M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, editors, *Automata, Languages and Programming*, volume 4052 of *Lecture Notes in Computer Science*, pages 144–155. Springer Berlin Heidelberg, 2006.
43. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In H. Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer Berlin Heidelberg, 2010.
44. T. Matsumoto and H. Imai. Public quadratic polynomial-tuples for efficient signature verification and message-encryption. In *EUROCRYPT*, volume 330 of *Lecture Notes in Computer Science*, pages 419–553. Springer, 1988.
45. R. J. McEliece. A Public-Key Cryptosystem Based On Algebraic Coding Theory. *Deep Space Network Progress Report*, 44:114–116, 1978.
46. D. McGrew and M. Curcio. Hash-Based Signatures. Request for Comments. Internet Engineering Task Force, 2014. Internet-Draft.

47. C. A. Melchor, P. Cayrel, P. Gaborit, and F. Laguillaumie. A new efficient threshold ring signature scheme based on coding theory. *IEEE Transactions on Information Theory*, 57(7):4833–4842, 2011.
48. A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone. *Handbook of applied cryptography*. CRC press, 2010.
49. R. C. Merkle. *Secrecy, Authentication and Public Key Systems*. PhD thesis, Stanford University, 1979.
50. R. C. Merkle. A Certified Digital Signature. In G. Brassard, editor, *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 218–238. Springer, 1989.
51. D. Micciancio. The hardness of the closest vector problem with preprocessing. *Information Theory, IEEE Transactions on*, 47(3):1212–1215, Mar 2001.
52. V. Miller. Use of elliptic curves in cryptography. In H. Williams, editor, *Advances in Cryptology Ů CRYPTO Š85 Proceedings*, volume 218 of *Lecture Notes in Computer Science*, pages 417–426. Springer Berlin Heidelberg, 1986.
53. R. Misoczki, J. Tillich, N. Sendrier, and P. S. L. M. Barreto. MDPC-McEliece: New McEliece variants from moderate density parity-check codes. In *Proceedings of ISIT*, pages 2069–2073. IEEE, 2013.
54. M. A. Morrison and J. Brillhart. A method of factoring and the factorization of F7. *Mathematics of Computation*, 29(129):183–205, 1975.
55. J. Patarin. The oil and vinegar signature scheme. Dagstuhl Workshop on Cryptography, September 1997.
56. E. Persichetti. Secure and anonymous hybrid encryption from coding theory. In *PQCrypto*, volume 7932 of *Lecture Notes in Computer Science*, pages 174–187. Springer, 2013.
57. C. Peters. Information–set decoding for linear codes over F__q. In N. Sendrier, editor, *Post-Quantum Cryptography*, volume 6061 of *Lecture Notes in Computer Science*, pages 81–94. Springer Berlin Heidelberg, 2010.
58. A. Petzoldt, S. Bulygin, and J. Buchmann. CyclicRainbow – a multivariate signature scheme with a partially cyclic public key. In *INDOCRYPT 2010*, volume 6498 of *LNCS*, pages 33–48. Springer, 2010.
59. A. Petzoldt, S. Bulygin, and J. Buchmann. Linear recurring sequences for the UOV key generation. In *Proceedings of PKC'11*, volume 6571 of *Lecture Notes in Computer Science*, pages 335–350. Springer, 2011.
60. A. Petzoldt, S. Bulygin, and J. Buchmann. A multivariate threshold ring signature scheme. *AAECC*, 2012.
61. A. Petzoldt and J. Ding. Gui – a fast HFEv- based multivariate signature scheme. to Appear.
62. C. Pomerance. The quadratic sieve factoring algorithm. In T. Beth, N. Cot, and I. Ingemarsson, editors, *Advances in Cryptology: Proceedings of EUROCRYPT 84, A Workshop on the Theory and Application of of Cryptographic Techniques, Paris, France, April 9-11, 1984, Proceedings*, volume 209 of *Lecture Notes in Computer Science*, pages 169–182. Springer, 1984.
63. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, STOC '05, pages 84–93, New York, NY, USA, 2005. ACM.
64. R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.
65. P. Rogaway and T. Shrimpton. Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance. In B. K. Roy and W. Meier, editors, *Fast Software*

Encryption, 11th International Workshop, FSE 2004, Revised Papers, volume 3017 of *Lecture Notes in Computer Science*, pages 371–388. Springer, 2004.

66. K. Saeedi, S. Simmons, J. Z. Salvail, P. Dluhy, H. Riemann, N. V. Abrosimov, P. Becker, H.-J. Pohl, J. J. L. Morton, and M. L. W. Thewalt. Room-Temperature Quantum Bit Storage Exceeding 39 Minutes Using Ionized Donors in Silicon-28. *Science*, 342(6160):830–833, 2013.
67. K. Sakumoto, T. Shirai, and H. Hiwatari. Public-key identification schemes based on multivariate quadratic polynomials. In *CRYPTO*, volume 6841 of *Lecture Notes in Computer Science*, pages 706–723. Springer, 2011.
68. P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.
69. D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In K. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 27–47. Springer Berlin Heidelberg, 2011.
70. J. Stern. A new identification scheme based on syndrome decoding. In *CRYPTO*, volume 773 of *Lecture Notes in Computer Science*, pages 13–21. Springer, 1993.
71. C. Tao, A. Diene, S. Tang, and J. Ding. Simplematrix scheme for encryption. In *PQCrypto*, volume 7932 of *Lecture Notes in Computer Science*, pages 231–242. Springer, 2013.
72. X. Wang and H. Yu. How to Break MD5 and Other Hash Functions. In R. Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 19–35. Springer, 2005.
73. Wikipedia. Timeline of quantum computing — wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Timeline_of_quantum_computing&oldid=613219069`, 2014. [Online; accessed 25-September-2014].