Reports of crypto's death are greatly exaggerated *

Christopher Williamson

April 23, 2018

Abstract

Even people new to the study of cryptography are likely to have some kind of awareness that the potential rise of quantum computers poses a threat to cryptography in some sense. However, if the author's experience is typical, this awareness is likely to be vague. We give a brief introduction to the topic of post-quantum cryptography, aiming to paint a clearer picture of the situation that cryptography finds itself in. In contrast to the title (and the opinions of the author), this article does not intend to persuade anyone of the brightness of cryptography's future but to simply explain the vulnerabilities and potential ways forward. This article is also intended for students who have taken Bogdanov's cryptography course at the Chinese University of Hong Kong; thus certain topics have been emphasized or deemphasized in order to avoid redundancy.

1 Introduction

While proofs of security abound in the field of cryptography, the foundations frequently lie on unproven assumptions of computational hardness. From these foundations, security proofs take the form of reductions, i.e. we claim that some particular protocol is secure as otherwise a widely held assumption is false.

Of course, in some sense, this is a highly nonoptimal state of affairs. Unconditional proofs are always superior to conditional ones. But with an understanding that sometimes we cannot prove what we strongly believe (and the fact that applications cannot wait for these proofs to materialize), this situation can be seen as a part of the beauty of cryptography. The field's web of reductions mean that all vulnerabilities can effectively be studied by considering a relatively smaller set of problems. If decades pass without any significant progress towards contradicting computational hardness, then confidence builds that systems whose security is based on the assumptions will be secure.

Quantum computers are a new model of computation that challenge some of our hardness assumptions. Luckily, cryptography's web of reductions allows us to make precise claims about what we do and do not know. In this article, we will describe the sources of vulnerabilities that arise from quantum computers of sufficient size, the negative effects on classical cryptography that such computation will bring, and some of the promising ways to proceed with cryptography.

2 Sources of vulnerabilities

Vulnerabilities to classical cryptography come from the ability of quantum computers to solve certain problems faster than classical ones can. The extent of the speedup depends on the

*a Mark Twain reference, because "Introduction to post-quantum cryptography" is too boring. We also do not pretend to introduce all of the ideas, or even all of the important ones.

particular problem. We first mention the two quantum algorithms that are by far the most influential in motivating the need for post-quantum cryptography. Remarkably, it is feasible to understand the cryptographic implications of these algorithms without background in how these algorithms work. For this reason, we defer to other sources for in-depth exposition about how these algorithms work, allowing us to focus on the implications of quantum computers without going into quantum topics themselves.

Shor's algorithm In 1994, Shor [18] gave a quantum algorithm for factoring integers. Shor's algorithm runs in time roughly quadratic in the size of the input. Specifically, to factor the integer n (which is of input size $\log n$), the running time is $\tilde{O}(\log^2 n)$, where the $\tilde{O}$ hides factors that are logarithmic in $\log n$. The best classical algorithm, the general number field sieve, takes time $2^{\tilde{O}(\log^{1/3} n)}$. This method of Buhler, Lenstra, and Pomerance [9] builds upon work of Pollard, which was originally used to factor Fermat numbers (of the form $2^{2^m} + 1$ for some integer m).

Crucially for our purposes, Shor's algorithm can be used to solve the discrete log problem in comparable time. [1]

Grover's algorithm Suppose that we are given oracle access to a database of size n, perhaps that specify the values of some function $f : \{0,1\}^{\log n} \to \{0,1\}$. This problem is called database search, or simply function inverting. It should be intuitively clear that if we are promised that some input to f results in a 1, then in the worst case, we will have to query the oracle $\Theta(n)$ times to find an input x so that $f(x) = 1$. This intuition breaks down when quantum computation is available. In 1996, Grover [11] developed a quantum algorithm for solving this problem that takes $O(\sqrt{n})$ time. This result is essentially optimal in typical models[2] of quantum computation by a lower bound of Bennett, Bernstein, Brassard, and Vazirani [2].

Implications Broadly speaking, we will see that cryptosystems that can be readily attacked by Shor's algorithm should be considered insecure against a quantum adversary. This is due to the subexponential speedup exhibited by Shor. (To be precise, we should say that these systems may still be secure against a quantum computer, but only if we blow up the key sizes exponentially to completely unrealistic lengths.) Grover's algorithm, which demonstrates a polynomial speedup, has less severe effects. It is generally acceptable to double key lengths to attain the same level of security: if a classical algorithm requires $2^{|\text{key}|}$ steps to break a protocol that is based on the hardness of performing database search, then a quantum computer may take only $\sqrt{2^{|\text{key}|}} = 2^{|\text{key}|/2}$ steps. We can return to the previous level of security by doubling the key length.

The most crucial consideration at this point is whether any given hardness assumptions falls victim to Shor's algorithm. It directly follows from the fact that Shor's algorithm can be used to factor and solve discrete log efficiently that protocols using discrete log hardness assumptions (such as the DDH assumption) are in trouble. This already jeopardizes a large fraction of the protocols that we have encountered in this course[3]. LWE based assumptions, as introduced by Regev [16], form the other class of hardness assumptions that we utilize frequently. In the classical setting, the standard LWE assumption is (as is necessary for cryptographic applications) an average case hardness assumption. This assumption is randomly self-reducible meaning that proving its worst-case hardness is approximately equivalent to showing its average-case hardness.

[1] This appears to follow not in a black-box reduction from discrete log to factoring, but via knowledge of how Shor's algorithm works on the factoring problem.

[2] In addition to significantly stronger models, there is a slightly stronger model of quantum computation due to Aaronson [1] in which this problem can be solved in $O(n^{1/3})$ time.

[3] Not to mention RSA, which is one of the most famous public key encryption schemes.

Then, Regev showed that worst-case LWE is at least as hard as certain lattice problems that are conjectured hard, such as GapSVP.[4] No application of Shor's algorithm towards such lattice problems is known as of yet, and thus protocols with hardness stemming from LWE-based assumptions and their underlying lattice problem hardness assumptions, are secure for the time being. Thus we see that some, but not all, of our assumptions are problematic in a post-quantum world. We have seen many applications of LWE (which we will soon review), which may seem to alleviate much concern about quantum algorithms. However, since LWE is still an assumption, it is clearly preferable to develop other assumptions that are not known to be quantum vulnerable and build more cryptography upon them. This way we are not putting all of our eggs in a small number of baskets. To this end, we will later illuminate more hardness assumptions that are presently in a similar condition as LWE: safe (for now) against quantum computers.

3 Accounting for our familiar protocols

Before moving forward with new assumptions and new cryptography, let's first consider vulnerabilities with respect to the protocols that we have gained the most familiarity with in class. For the purposes of this brief article, we primarily consider implications to three of the most frequently utilized cryptographic primitives: key exchange, encryption, and signatures. We will first handle the symmetric key setting before setting it aside, as such protocols are less affected by quantum prospects.

Symmetric key cryptography In symmetric key settings, users Alice and Bob are assumed to already hold a private key known to them; security of key exchange protocols is not relevant here. With respect to encryption, we have already seen how to obtain an encryption scheme assuming that pseudorandom functions exist. This assumption is weaker than the validity of the discrete log or LWE assumptions: we have seen how PRFs can be constructed via these assumptions - and other constructions may exist even in the event that these assumptions are unfounded. Thus, we need only have confidence in the existence of PRFs in order to have a post-quantum secure encryption scheme, up to perhaps modest (constant factor) increases in key length to account for any possible applications of Grover's algorithm.

Message authentication codes are the symmetric key analog of signatures. Similarly to the case of encryption, MACs can be constructed from solely PRFs or a combination of PRFs and hash functions (both constructions were seen in class). Again, the existence of PRFs is a relatively weak assumption, and we have seen candidate collision-resistant hash functions based on LWE (and also on the quantum-vulnerable DDH assumption).

Public key cryptography In the public key setting, we have made extensive use of the assumption of the computational hardness of taking discrete logarithms. The Diffie-Hellman key exchange would likely be insecure in light of a quantum computer, although we have also seen the Ding-Lin key exchange, which depends on a variant of the LWE assumption.[5] One of the prominent encryption schemes that we have seen, El Gamal encryption, is also discrete log based and subject to quantum attacks. However, we did construct (in homework 2) an

[4]This problem is to determine whether the shortest vector in a lattice is "small" or "large", under the promise that the shortest vector is of magnitude at most 1 or at least β which is bounded away from 1.

[5]Recall that LWE postulates that the random variables (A, r) and $(A, Ax + e)$ are indistinguishable when A and x are random over elements of $\mathbb{Z}_q$ and e is a vector of random noise of bounded magnitude $b << q$. The shortLWE assumption is the variant where we assume indistinguishability holds even when x is taken over the "smaller" distribution from which the error e is sampled.

LWE-based (again, shortLWE) encryption scheme. We have seen how to apply the Fiat-Shamir heuristic to Schnorr's identification protocol to obtain digital signatures. However, the security of Schnorr's underlying identification scheme is discrete log based and the signatures are only proven secure in the random oracle model. Additionally, we did see how to construct signatures using just a pseudorandom generator[6]. This scheme was very inefficient in that it only allowed for a single message to be signed. There are extentions to such signatures, as we will see later. This area of "hash-based signatures" is considered appealing due to the lack of dependence on strong assumptions and the resulting post-quantum confidence.

4 New assumptions and new protocols

Now that we know where we stand, we introduce some new cryptography that is believed to be secure against quantum adversaries. To get a sense for which areas are most active, we take a look at the "competition" that has been set up by the National Institute of Standards and Technology (NIST), in the United States. The institute has solicited researchers from around the world for submissions of protocols for key exchange, encryption, and signatures that can be expected to be secure in a post-quantum environment. Below is a table showing the number of submissions that they received (submissions closed in November 2017).

	Key Ex./Enc	Sig
lattice-based	24	4
multivariate-based	6	7
hash-based	–	4
code-based	19	5
other	10	3

Table 1: Categorization of NIST submissions

We take these submissions as a guide, and discuss in varying levels of detail each of the four areas that constitute the majority of submissions.

4.1 Lattice-based cryptography

Although lattice-based cryptography is an extremely active area, we will only briefly discuss it so as to minimize redundancies with class material.
The definition of a lattice is very simple: it is the set of integer linear combinations of some set of basis vectors in $\mathbb{R}^n$. We briefly introduce some of the lattice based computational problems that are considered to be hard.

Shortest vector problem (SVP) The input of this problem is the basis of some lattice and a norm of choice (which is usually Euclidean). The problem is to find the non-zero lattice element that has the minimum norm. This problem is only known to be NP-hard for the infinity norm, although there are limited results for other norm choices (NP-hard for randomized or for quasipolynomial reductions). There is also an approximation version of the problem, where one must find a vector of norm within a fixed factor of the optimal norm.

Closest vector problem (CVP) For this problem, the input also includes a basis of a vector space V and a vector v in V. The goal is to find a lattice element that is closest to v, and this is NP-hard for all L_p norms.

[6]The construction seen in class is essentially Lamport's one-time signature scheme.

Gap versions The problems GapSVP_β and GapCVP_β are the relevant decision promise problems parameterized by $\beta > 1$. For GapSVP_β, one must decide whether the shortest lattice vector is of norm less than 1 or greater than β. GapCVP_β is defined in the obvious way and there are a variety of hardness results depending on the asymptotics of β.

Lattices and the natural problems that result from them are very influential in cryptography. Perhaps most notably, the LWE assumption bases its hardness on a particular instance of GapSVP that is presumed to be hard against even a quantum adversary. There are a zoo of lattice-based LWE assumption variants, such as ring-LWE, which is heavily used in post-quantum key exchange protocols. Ring-LWE is a stronger assumption and some recent work (such as [8]) addresses the problem of modifying protocols that depend on a strengthened version of LWE so that they depend only on the standard version.
In addition, many cryptographic protocols are built directly on top of lattice problems, without going through another assumption like LWE.

4.2 Multivariate-based cryptography

Hardness of these cryptosystems is based on solving systems of multivariate polynomial equations (this problem is known to be NP-hard and presumed hard for quantum computers). Frequently, the polynomials are taken to be of degree at most two, and the problem of solving systems of them are utilized towards signature schemes. We sketch a high-level example that can be seen as a template of the 1999 "unbalanced oil and vinegar" scheme of Paturin.

In this signature scheme, the public key is a set of particularly chosen quadratic polynomials $f_1, ..., f_m$ over variables $x_1, ..., x_n$. To sign a message $y = (y_1, ..., y_m)$, the signer sends y and the values $x_1, ..., x_n$ so that $y_i = f_i(x_1, ..., x_n)$ for all $i = 1, ..., m$. The general problem is presumed hard on average (even against a quantum adversary) when $n \approx m$. The trick of these schemes is to insert a hidden structure into the choice of the f_i so that the signer can efficiently create signatures.

4.3 Hash-based signatures

Hash-based cryptography centers around the use of hash functions to create signature schemes based on very minimal assumptions, namely the existence of sufficiently good hashes. We have in class already seen an example of this (although stated with respect to a PRG instead of a hash). The example from class is typically called Lamport's one-time signature. In this scheme, the signer randomly picks a pair of inputs X_0, X_1 from the domain of a collision resistant hash function H. These values serve as the secret key. The public key is set to Y_0, Y_1 where $Y_b = H(X_b)$ for $b = 0, 1$. To sign a single bit message b, the signer simply appends the value X_b.

Of course, this signature scheme is very limiting in that it can only be used to produce a single signature. A trivial way to allow for multiple signatures is to generate many independent key pairs, publish a long public key, and only use a fraction of the public key for each signature. However, there is a way to avoid this linear (in the number of signatures) blow-up in the public key length using something called a Merkle tree.

Suppose that the signer has a desire to sign N times where N is a power of two. The signer can produce N key pairs for the one-time Lamport scheme and "place" these as leaf nodes in a complete binary tree with N leaves[7]. Then, the hash function is applied along the edges of the tree, so that two key pairs are hashed and this value is placed on their parent node, and so

[7]Note that we still have to generate many key pairs, although we will see that the single public key is shorter than one obtains via trivial repetition.

forth continuing until the root node is reached. The value of the root node is then published as the public key. To sign the ith signature, the signer uses the key pair at the ith leaf to sign a standard one-time Lamport signature. Then, the signer appends to this the necessary hash values at intermediate nodes to allow the receiver to reconstruct the public key in his verification phase. The use of a collision resistant has function means that someone trying to forge a signature must be able to find inputs to H so that the result is the public key. This is a contradiction of the properties of H.

Thus we have seen two versions of the Lamport signature: the original version and one that allows for multiple signatures. Unfortunately, both schemes are still *stateful*, in that the signer must keep track of how many signatures have been generated.

Minimality of assumptions and relevance to post-quantum cryptography Despite that hash-based signatures date from the 1970s, their relevance is arguably increasing today as the desire to develop quantum safe primitives sharpens. These signatures depend only on the existence of sufficiently good hash functions, a relatively weak assumption considering that many of our present-day assumptions would already imply the existence of such functions (and hashes may exist even if these assumptions fail). In fact, as long as one is ready to presume that signature schemes exist at all, then hash-based signatures are not really an assumption: Rompel [17] showed in 1990 that hash-based signatures are implied by the mere existence of signatures[8]. Additionally, recent work has given hash-based signatures that are not stateful. One such example is the NIST submission SPHINCS+ of Bernstein et al [7].

4.4 Code-based cryptography

We first review basic details about linear error-correcting codes and then introduce an encryption scheme that is based on the hardness of their decoding.

Linear error-correcting codes A linear code of alphabet size q is a subspace C of $\mathbb{F}_q^n$. Codewords consist of elements of this space. For our purposes, we presume that $q = 2$, i.e. that we have a binary linear code. If the dimension of C is k, then there exists a set of k basis vectors in C such that every codeword is a linear combination (or sum when $q = 2$) of basis vectors. In this way, every subset of the k basis vectors yields a distinct codeword, and thus the message space is of size 2^k; the coding scheme maps messages of length k to codewords of length n. If the distance (in the Hamming metric) of any two codewords is at least $2t + 1$, then we have that the code can correct up to t errors. We denote this as a $[n, k, t]$-code[9]. Concatenating the k basis vectors in $\mathbb{F}_2^n$ forms what is a $k \times n$ generator matrix G for the code C. Codewords for the message m may be formed by computing mG.

4.5 The McEliece encryption scheme

This scheme was introduced by McEliece in 1978 [13]. We first give the definition of the scheme under a generic binary linear code. The scheme is then instantiated with a particular choice of code.

Alice has an arbitrary binary linear $[n, k, t]$ error-correcting code C with $k \times n$ generating matrix G. The encryption scheme consists of the following three algorithms:

[8] In fact, he showed that the existence of one-way functions is logically equivalent to the existence of signatures in general.

[9] Frequently this would be called a $[n, k, 2t + 1]$-code where the third parameter gives the distance between code words instead of the correction capability.

- *Key Generation.* She samples a uniformly random binary matrix S that we require to be $k \times k$ and of full rank. She also picks a uniform $n \times n$ permutation matrix P.[10] She sets her secret key as (S, G, P), and she computes $G' = SGP$ to obtain her public key (G', t).
- *Encryption.* Bob has message $m \in \{0,1\}^k$. He treats G' as a generating matrix for a linear code and computes $c' = mG'$. He then outputs $c = c' + z$ where $z \in \{0,1\}^n$ is a uniformly random string of Hamming weight exactly t.
- *Decryption.* To decrypt c received from Bob, Alice computes cP^{-1} and applies the decoding algorithm for code C to it, obtaining m'. She then outputs $m'S^{-1}$.

Functionality To check that Alice correctly decodes a message from Bob, we want to ensure that $m'S^{-1} = m$. Note that Alice receives c and computes:

$$cP^{-1} = (c' + z)P^{-1} = mG'P^{-1} + zP^{-1} = mSG + zP^{-1}$$

P, and thus P^{-1} are permutation matrices, and therefore the Hamming weight of zP^{-1} is the Hamming weight of z, i.e. t. The quantity $mSG+zP^{-1}$ is of the form of a codeword for message mS under code C with at most t errors; thus the decoding algorithm returns mS as the message associated with the noisy codeword. Alice's multiplication by S^{-1} then results in m, as desired.

Instantiation As described, the McEliece encryption wraps around some choice of linear code. While several choices have been considered, the use of Goppa codes has appeared to be the most robust choice. The details of this coding scheme are out of the scope of this article, but there are an abundance of resources about them [5]. Decoding proceeds by the Patterson algorithm [15], which runs in time polynomial in n. Bernstein [6] also gives an improved decoding algorithm.

Security Security for the McEliece scheme is conjectured due to the hardness of decoding a general linear code, which is known to be NP-hard [4]. Thus, the reason that McEliece appears to be secure is that it is not known how to exploit the fact that the scheme uses a randomized Goppa code instead of using a truly random linear code. This scheme is also presumed to be secure against quantum adversaries as no polynomial-time quantum algorithm is known for decoding general linear codes.
Note that the scheme as presented is OW-CPA but not even semantically secure. There are variants that exist to upgrade the security to IND-CCA2 (confer [14], [10]). Bernstein et al have submitted a McEliece based key exchange mechanism to NIST[11].

4.6 Quantum cryptography

It is common to see warnings about the conflation of post-quantum cryptography and quantum cryptography (confer wikipedia). Granted, quantum cryptography is clearly a different animal from the cryptography that we have been studying; its foundations lie on the physics surrounding quantum mechanics rather than the computational hardness of any mathematical problems. However, this distinction means that such protocols are often information-theoretically secure. This is the main justification we use for including a brief paragraph about it here: information theoretic cryptographic protocols are indeed post-quantum secure - in fact they are secure against any form of computation.

[10] Permutation matrices are those that have a single 1 in each row and column and 0s elsewhere. They operate on a binary vector as a permutation.

[11] Their protocol actually uses something called Niederreiter encryption, which is a "dual" version of the McEliece scheme.

The first quantum cryptography protocol was a 1984 key exchange system of Bennett and Brassard [3]. The unconditional security of the protocol stems from the quantum mechanical "no cloning" theorem.

Despite this power, Mayers gave an interesting negative result in 1997 [12], namely that unconditionally secure quantum commitment protocols do not exist. However, there exist other models of quantum computation that allow for quantum commitments.

5 Conclusions

Post quantum cryptography may be more familiar than the uninitiated might expect. The new sources of vulnerabilities that quantum computers introduce are, for now, largely confined to the effects of the algorithms of Shor and of Grover, although only the former appears to have the potential to completely eliminate some of our classical hardness assumptions. Although Shor's algorithm may eventually make obsolete factoring and discrete log based protocols, the wealth of cryptography that is LWE-based is still considered secure. Additionally to LWE, there are many schemes that are based on other lattice assumptions, or from completely different assumptions altogether. We have seen some of these alternate systems that are multivariate or code based. There is also a surprising amount of cryptography that can be based on the very lightweight assumptions of good hash functions, pseudorandom generators, and pseudorandom functions. Quantum cryptography offers a different approach and the field reminds us that any of our classical cryptography[12] that is information theoretically secure is still intact.

References

[1] Scott Aaronson. Quantum computing and hidden variables. *Physical Review A*, 2005.

[2] CH Bennett, E Bernstein, G Brassard, and U Vazirani. Strengths and weaknesses of quantum computing. 1997.

[3] Charles H Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. *Theor. Comput. Sci.*, 560(P1):7–11, 2014.

[4] Elwyn R Berlekamp, Robert J McEliece, and Henk CA van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.

[5] Daniel J Bernstein. Introduction to post-quantum cryptography. In *Post-quantum cryptography*, pages 1–14. Springer, 2009.

[6] Daniel J Bernstein. List decoding for binary goppa codes. In *International Conference on Coding and Cryptology*, pages 62–80. Springer, 2011.

[7] Daniel J Bernstein, Daira Hopwood, Andreas Hülsing, Tanja Lange, Ruben Niederhagen, Louiza Papachristodoulou, Michael Schneider, Peter Schwabe, and Zooko Wilcox-OHearn. Sphincs: practical stateless hash-based signatures. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 368–397. Springer, 2015.

[8] Joppe Bos, Craig Costello, Léo Ducas, Ilya Mironov, Michael Naehrig, Valeria Nikolaenko, Ananth Raghunathan, and Douglas Stebila. Frodo: Take off the ring! practical, quantum-secure key exchange from lwe. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1006–1018. ACM, 2016.

[12] i.e. many secret sharing schemes, multiparty computations, etc.

[9] Joe Buhler, Hendrik Lenstra, and Carl Pomerance. Factoring integers with the number field sieve. In Arjen Lenstra and Hendrik Lenstra, editors, *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*, pages 50–94. Springer-Verlag, 1993.

[10] Nico Dottling, Rafael Dowsley, Jörn Muller-Quade, and Anderson CA Nascimento. A cca2 secure variant of the mceliece cryptosystem. *IEEE Transactions on Information Theory*, 58(10):6672–6680, 2012.

[11] Lov Grover. A fast quantum mechanical algorithm for database search. *J. of the ACM*, 1996.

[12] Dominic Mayers. Unconditionally secure quantum bit commitment is impossible. *Physical review letters*, 78(17):3414, 1997.

[13] Robert McEliece. A public-key cryptosystem based on algebraic coding theory, 1978. DSN Progress Report 42-44.

[14] Ryo Nojima, Hideki Imai, Kazukuni Kobara, and Kirill Morozov. Semantic security for the mceliece cryptosystem without random oracles. *Designs, Codes and Cryptography*, 49(1-3):289–305, 2008.

[15] N. Patterson. The algebraic decding of goppa codes. In *IEEE Transactions on Coding Theory*, pages 52–66. IEEE, 1975.

[16] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, 2005.

[17] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 387–394. ACM, 1990.

[18] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.