Post Quantum Solutions outweigh the Threats to Security against Quantum Computers

Alexander Lindenbaum

Department of Computer Science, Columbia University

COMS 6998: Reading the CS Classics

Prof. Christos Papadimitriou and Jason Milionis

May 8, 2023

1. Introduction

Recently, the technological advancement and cultural popularization of quantum computing has brought to light its potential threat to public key cryptography. Indeed, theorists have worried about this threat since Shor's breakthrough paper (1999), and maybe even earlier. But now that corporations/governments are amassing resources and building quantum computers, it is high-time to ask whether or not we should be worried about losing privacy to entities with concentrated computing power. What is the nature of the quantum versus cryptography dynamic, how can we mitigate the damage quantum computers may deliver to our popular cryptographic protocols, and will we mitigate the risks before a worst-case scenario?

I believe that there is evidence for security in a world where quantum computers exist and perform without error. Justifying my belief involves returning to the roots of public key cryptography and quantum algorithms. This paper will then delve in current work on *post quantum cryptography*, the most successful endeavor to protect message privacy from quantum eavesdroppers.

Note that the details of how quantum computers work is beyond the scope of the paper; I wish to focus on how quantum interacts with cryptography.

1. Public Key Cryptography and Hardness Assumptions

Up until the late 1970s, all schemes for encrypting and decrypting secret messages were in the form of *symmetric key* cryptography. Suppose Alice wished to encrypt a message to send to Bob, such that any eavesdroppers would have a low chance of correctly decrypting the message. In order to do so, Alice would have to generate a secret key and use that key to encrypt the message. Bob would likewise need the secret key to decrypt, and the key would have to be transmitted in a secure channel (typically in person). One standard protocol for achieving security was the *One-Time Pad* (OTP). While OTP guarantees security in symmetric key cryptography, it exhibits two key issues: (1) the size of the secret key is as large as the size of the message to be encrypted, and (2) security is not guaranteed if Alice uses the same secret key to send two or more messages. There were other protocols which could mitigate these issues to some degree. But symmetric key cryptography naturally has a weakness that the secret key must be delivered in person. As such, encryption over the Internet would not be possible using exclusively symmetric key schemes.

Diffie and Hellman's groundbreaking paper, *New Directions in Cryptography* (1976), introduced *public key*, or *asymmetric*, cryptography, which made it possible to have secure message transmission without having to initially share a secret key. In these schemes, the receiver generates a pair of keys: a secret key (which the receiver keeps and does not reveal), and a public key. The public key is sent to the sender, who uses it to encrypt the message. The secret key decrypts. The key generation is such that the two keys are linked, and an adversary must guess the right corresponding secret key in order to correctly decrypt.

The prevailing public key cryptographic schemes, like RSA and elliptic curve, are based on hardness assumptions from theory. One can design a scheme so that provably, if an adversary could break the scheme faster than in exponential time, then that adversary could also solve some computational problem faster than in exponential time. If that computational problem is one which has evidence of being a hard problem to compute, then the cryptographic scheme has a *conditional* guarantee of being secure (breaking the scheme implies solving a hard problem which we believe is not possible).

Diffie and Hellman's protocol was based on the problem of computing discrete logarithms (DL) for carefully chosen groups (1976). Shortly after their paper, RSA was devised, based on the difficulty of factoring large numbers (FACTORING) (Rivest et al., 1978). More important protocols include ones based on elliptic curves — introduced separately by Miller and Koblitz (Miller, 1986), (Koblitz, 1987) — and a key exchange protocol called Buchmann-Williams, named after the two authors (1988). These protocols are now used in government, industry, and over the Internet, and are celebrated as major achievements in computer science and cryptography.

Whereas OTP is not based on a hardness assumption, all of the above mentioned protocols are, and this translates to a different guarantee of security. Problems like DL and FACTORING are in NP. Conditional security arises because we believe that there are no polynomial time algorithms for DL or FACTORING (or at least, finding such an algorithm is not easy). But any computer that can solve FACTORING would break the RSA scheme as a consequence. So the security of Diffie-Hellman and RSA is dependent on our assumptions for the computational limits of the eavesdropper.

2. Quantum Undermines Public Key Cryptography

The idea that quantum phenomena can be exploited for purposeful computation came out of the problem of quantum simulation, but generally became the question: for which problems does quantum give some kind of leverage? Before the first quantum computer was physically built, theorists were considering algorithms for quantum computers, and several interesting speedups were discovered. Most importantly for the field of cryptography, Peter Shor gave an algorithm that solves FACTORING and DL in polynomial time (1999). Quickly, the paper became popular and surprised cryptographers: an adversary equipped with a quantum computer could easily break schemes like RSA, which are standard for many applications. Other such quantum algorithms followed eventually. For example, one can solve Pell's equations, which breaks Buchmann-Williams key exchange (Hallgren, 2007).

In order to break an RSA scheme with a secret key of n bits, the quantum computer requires n qubits. The reason for this is that the final output of a quantum algorithm is typically the result of measuring all qubits, after gates are applied to them. Each qubit reading gives one bit, and you need n to describe the secret key, hence the need for n qubits. Typical uses of RSA have a secret key of 1024 or more bits.

Where are we now in terms of the number of qubits in a quantum computer? The first quantum computer was built in 1998 and held two qubits (Chuang et al., 1998). Progress on increasing the number of qubits in a computer has been slow, as many resources are required to maintain low energy states for the qubit particles. Recently, IBM unveiled their "Osprey" project with allegedly 433 qubits (PRNewswire, 2022). The company has not yet released specs on the noise rate, which may result in faulty algorithms unable to compute Shor's algorithm.

Stimulated by the results of Shor and others, and by improving technology, more effort has been extended towards two studies — and opportunities — to mitigate the damage quantum computers *might* deliver to security. One is *quantum cryptography* (QC); the other is post *quantum cryptography* (PQC). In QC, we give the power of quantum computers to the sender/receiver as well as the eavesdropper. In PQC, the cryptographic protocol is classical but resistant to quantum attackers.

3. An Interlude into Quantum Cryptography

Quantum cryptography began with Wiesner's paper *Conjugate Coding*: his result was the first evidence that (1) quantum can encrypt and decrypt, and (2) quantum gives security against stronger-than-polynomial time adversaries (1983). How the protocol utilizes quantum phenomena is beyond the scope of this paper, but the effect was for two parties to swap messages encoded in different qubit phases, such that no eavesdropper could recover the message. Just like OTP, the security of this protocol is *unconditional*, meaning that security is guaranteed independent of the adversaries. Wiesner also in *Conjugate Coding* introduced a scheme for "quantum money" in which bank notes come equipped with a series of qubits encoding your identity and some certifications. According to Scott Aaronson, *Conjugate Coding* was the first paper to suggest that qubits operate distinctly from classical bits, and that leads to new cryptographic uses in particular (2021).

Wiesner's protocol was eventually adapted into the BB84 protocol, due to Bennett and Brassard (Perlner & Cooper, 2009). BB84 is particularly useful for the key distribution problem in cryptography: how to send a secret key to use for future encryption, without an already established secure channel of communication. The protocol is an implementation of OTP that transmits qubits, as opposed to classical bits. Security is also proven theoretically unconditional, although there have been successful attacks on BB84 by tampering with the infrastructure necessary to execute the protocol.

One benefit of BB84, and the improvements which succeeded BB84, is that not only do we have theoretical security against adversaries with quantum computers, but tampering with the protocol will result in the sender and receiver gaining knowledge that an attack was attempted! Security and this additional property stem from the *no-cloning theorem* in quantum mechanics, which demonstrates that the state of a qubit cannot be perfectly copied. An attacker would have to copy either the plaintext or the ciphertext in an attack, which would in turn alter the original message and alert the sender.

However, there are clear downsides to resorting to quantum computers for encryption/decryption. For one, both “Conjugate Coding” and BB84 require infrastructure for transmitting qubits over distances, and for devices to be able to store/manipulate these qubits. Moreover, any interference from the environment introduces noise into our protocols, which for some methods renders them useless. Perhaps the most serious implication of using quantum for cryptography is that every person would require a quantum processor. We are very far from implementing such a mass infrastructure.

However we should consider quantum technologies for security somewhat overkill, considering the fact that many of our classical paradigms are still resistant to quantum attacks. Hash-based techniques, which are used to construct digital signature schemes and more, still require quantum computers to “guess” the secret, which is the best that we can hope for in security (Perlner & Cooper, 2009). The most popular of such methods today is Merkle hashing, introduced in Merkle’s seminar paper (Merkle, 1979). Merkle hashes are also the predominant hashing technique used in zero knowledge proofs and security in blockchains, as they are essential to constructing polynomial interactive oracle proofs (Thaler, 2022). These schemes will not go away if quantum computers become more powerful and accessible, although replacing RSA may still be necessary.

Another interesting development in quantum cryptography is the use of quantum to solve other related problems in cryptography. I mention one (because it has benefited from extreme improvements in the quantum world), dubbed *oblivious transfer* (OT) and first introduced by Rabin (1981). The problem of OT is for the sender to be able to send two messages to the receiver — not necessarily encrypted — but encoded in such a way that the receiver can only open one of the two: not both. On top of that, the sender should not learn any information about which secret the receiver opened. Over the years, we have slowly improved the guarantees we can get from algorithms for OT, and recently it was shown that secure OT can be used to construct a secure protocol for multiparty computation (MPC), and with an efficient reduction (Ishai et al., 2008). In MPC, *n* people collectively wish to compute a function of their private information, without revealing any of their data to the other participants. For some functions, it is easy to achieve secure MPC. Secure OT gives secure MPC for all functions.

In the quantum world, the same question could be asked about OT and MPC, but proving formal guarantees is much more difficult. In a breakthrough result, it was shown that if quantum-hard one-way functions exist, then OT is secure against quantum adversaries (Bartusek et al., 2021). Because of the 2008 result, this immediately implies that we would have secure MPC against participants with quantum computers as well. Thus figuring out secure multiparty computation reduces to constructing functions which are easy to evaluate, but not even quantum computers can invert them on average.

There are other advances and open questions in the realm of quantum cryptography, not mentioned here. Because it is currently unrealistic to consider giving every person a quantum processor, quantum cryptography is often shadowed by the study of post quantum cryptography. The solutions in PQC are relatively easier to implement, and no mass hardware overhaul is necessary to achieve post quantum security.

4. Post Quantum Cryptography

The quandary within classical cryptography is that unconditionally secure methods are inherently cumbersome for the users, and the hardness conditions of public key cryptography are broken by quantum computers. In order to keep public key cryptography, we thus need to come up with new hard computational problems to base such schemes on. But those problems should be difficult for quantum computers to solve as well as classical ones. PQC attempts to find such candidates and give evidence for their hardness.

One factor which has led to PQC flourishing has been support from the US government, in the form of the National Institute of Standards and Technology (NIST). NIST gives grants to PQC research and regularly hosts competitions (Alagic et al., 2019). They have since 2016 asked cryptographers to submit PQC candidates and test their security.

FACTORING and DL are NP-intermediate problems: in NP but not believed to be NP-complete. If they are broken by quantum computers, why not base cryptographic schemes on NP-complete problems like SAT or integer programming? Taking it to the extreme, why not base the scheme on a PSPACE-complete problem? We cannot use a PSPACE-complete problem because we require that the protocol be efficiently executable by the sender/receiver. For the former, we have some theoretical results to suggest infeasibility. Bogdanov and Trevisan showed that if the security of a one-way function is based on a NP-complete problem being hard to solve, then the polynomial hierarchy collapses (2006). One-way functions are a fundamental building block of cryptography and exist if and only if many other cryptographic primitives exist. The polynomial hierarchy categorizes increasingly harder complexity classes: most believe that its collapse is unlikely. All this goes to show that NP-intermediate problems are the sweet spot for hard computational problems.

The remaining question is: what NP-intermediate problems are also hard for polynomial time quantum computers? It may be the case that no problems in NP are hard for polytime quantum, but this is unlikely. Randomly choosing a NP language, with probability 1 no quantum machine equipped with the problem oracle can accept the language (Bennett et al., 1997). This does not prove that BQP is a strict subset of NP, but gives some very strong intuition for the case. Unfortunately, proving that any particular problem is hard for quantum computers seems to be a very difficult task, just as in the classical case. Nevertheless, researchers have provided some candidate problems that seem to be hard even in the quantum world. Lattice problems are among the most well-studied candidates.

Imagine you are in n-dimensional real space, and you have a list of n nonzero vectors, called a *basis*. The set of all *integer* combinations of those n vectors is a *lattice*. Given as input a basis for a lattice, would you be able to compute particular properties of that lattice? For example, while we know that 0^n is always in the lattice, what is the shortest *nonzero* vector in the lattice? This problem is known as the *shortest vector problem* (SVP) (Peikert, 2016). There are several other related problems, not mentioned here. An important subclass of these lattice problems are the approximation problems. GapSVP_γ gives you a basis for a lattice in n dimensions and asks: does the shortest vector have length less than 1 or length greater than $\gamma(n)$? This problem is particularly important for lattice cryptography: GapSVP, where $\gamma(n)$ is polylogarithmic in n, is a problem which lies in the intersection of NP and coNP (Aharonov & Regev, 2005), making it very unlikely to be NP-complete (unless NP = coNP and the polynomial hierarchy collapses). This makes it a great candidate to base cryptographic security upon, and indeed many of the schemes have hardness assumptions that eventually reduce to GapSVP.

As an aside, the first lattice-based public key cryptographic scheme, NTRU, has not been proven to be related to any lattice problems (Peikert, 2016). Due to Hoffstein et al., NTRU is a ring-based system which roughly translates to working over algebraically structured lattices (1998). Only special cases of NTRU have been shown to reduce to hard problems, yet it still celebrates concrete security against many years of cryptanalysis.

Perhaps the most celebrated problem which directly translates into a public key cryptographic scheme is called the *Learning with Errors* (LWE) problem. LWE is expressed as a linear algebra problem and has a very nice lattice-based interpretation. LWE is posed as such: you are given a matrix A and a vector b such that there is some x where Ax = b. Given only (A, b), find the x that satisfies the equality.

As is, this problem is easy, even for classical computers. It requires a simple application of Gaussian elimination, which runs in O(n^3) time. So we modify the problem to make it more difficult to solve, in the following way: before (A, b) is given, modify b by adding some small integral Gaussian error to each entry to obtain b'. The goal is still to find x such that Ax = b (not b'). Now if we try to use Gaussian elimination, the errors accumulate exponentially, and the x we recover probably does not satisfy the equation. All we know is this: (1) A and b have integer-valued entries and all operations are modulo n (meaning that x also has integer values), and (2) b' is some vector very close to b, but it's unsure which integer valued vector. It is believed that this problem is computationally hard, even for quantum computers.

Public key protocols have been built on top of this question, but is LWE provably hard? The answer to this question is in the same paper that introduced LWE, due to Oded Regev (2009). He proved that solving LWE is at least as hard as solving GapSVP. Not only that, but it is at least as hard as quantumly solving GapSVP: meaning that the quantum algorithm has query access to a black-box LWE solver. Since GapSVP is believed to be a hard problem for even quantum computers, we can extend that hardness to LWE.

Another important lattice problem used for cryptography is the *Shortest Integer Solution* (SIS) problem, first introduced by Ajtai (1996). SIS can be thought of as the dual to LWE: given a random m x n integer-valued matrix A, find an x such that Ax = 0^n. Again, alone this problem is trivial (choose x = 0^m). So we demand that x is the shortest, integer, nonzero solution to the system. Given A, we can in polynomial time find a basis for its nullspace. But out of the nullspace we must pick an integer-valued nonzero solution, and it must be the shortest out of all solutions (Peikert, 2016). We believe that this problem is hard, because it also reduces to GapSVP (although this reduction is not quantum). Both SIS and LWE are studied intensely: after a decade of scrutinous cryptanalysis many believe either of these problems to be viable post quantum cryptography candidates.

Possibly the best argument we have that NTRU/LWE/SIS are hard for quantum computers is simply that we have not yet found a quantum algorithm for any of the problems. Similarly to the P vs NP question, this gives good intuition as to why lattice cryptography is quantum secure, but is not a valid proof. Researchers also argue that these problems are harder as they move away from number theoretic problems, which are more prone to having randomized solutions. A successful randomized attack on a cryptographic scheme is sufficient to break it, and we base the strength of a scheme off of defense versus classical/quantum computers' randomized attacks. For the three lattice problems mentioned above, we have not even found randomized, quantum solutions! That being said, all it takes is one solution to refute the idea that lattice problems give security.

So how easy is it to implement the LWE-based public key scheme, as opposed to RSA? The LWE scheme (encryption and decryption) requires a constant number of matrix vector multiplications, which can be done in polynomial time. RSA requires n group operations. Both seem to use up a similar amount of time. LWE does however require more space for the public key/secret key pair than RSA. There are variants of LWE, such as ring-LWE (Peikert, 2016), which impose some structure on the A matrix for the purposes of compression while still yielding conditional security.

5. Conclusion

We return to the question: should we be worried about the security threats quantum computers may pose in the close future? In an ideal world, all of our encryption would be swiftly replaced with post quantum encryption, and that LWE or any other PQC candidate would indeed be secure against quantum attackers. This way forward, as opposed to implementing the infrastructural changes necessary for quantum cryptography, seems much more feasible and cost-effective. There is one major hurdle to standardizing LWE-based encryption which we have yet to confront, which is the difficulty in bringing LWE to every router on the Internet. Such a task is not a trivial one.

We must also consider how much time we have before quantum computers are sufficiently powerful enough to undermine current-day cryptographic security. As mentioned before, IBM reports to have the computer with the most number of qubits: 433 (PRNewswire, 2022). However, this does not mean that “Osprey” can break RSA with a secret key of size at most 433 bits. The reason for this is largely noise in these computers. Perfect execution requires that the qubits not entangle with any external particles, and the qubits cannot receive any energy from external sources. In particular, Shor's algorithm for FACTORING employs the *Quantum Fourier Transform* (QFT), a gate which is heavily susceptible to noise and errors easily (Kitaev, 1997). Until quantum computers can totally isolate qubits (if even possible), or till theorists find noise-tolerant solutions to RSA, our current modes of encryption are safe. Thus the question of security against quantum will be resolved in this arms race: will noise be corrected in quantum computers before post quantum cryptography is mass-implemented? I believe that the latter is likely to precede the former, and that PQC is truly secure against quantum.

References

Aaronson, S. J. (2021). Stephen Wiesner (1942-2021). Shtetl-Optimized. Retrieved from https://scottaaronson.blog/?p=5730.

Aharonov, D., & Regev, O. (2005). Lattice problems in NP ∩ coNP. Journal of the ACM (JACM), 52(5), 749-765.

Ajtai, M. (1996, July). Generating hard instances of lattice problems. In Proceedings of the twenty-eighth annual ACM symposium on Theory of computing (pp. 99-108).

Alagic, G., Alperin-Sheriff, J., Apon, D., Cooper, D., Dang, Q., ... & Smith-Tone, D. (2019). Status report on the first round of the NIST post-quantum cryptography standardization process.

Bartusek, J., Coladangelo, A., Khurana, D., & Ma, F. (2021). One-way functions imply secure computation in a quantum world. In Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part I 41 (pp. 467-496). Springer International Publishing.

Bennett, C. H., Bernstein, E., Brassard, G., & Vazirani, U. (1997). Strengths and weaknesses of quantum computing. SIAM journal on Computing, 26(5), 1510-1523.

Bogdanov, A., & Trevisan, L. (2006). On worst-case to average-case reductions for NP problems. SIAM Journal on Computing, 36(4), 1119-1159.

Buchmann, J., & Williams, H. C. (1988). A key-exchange system based on imaginary quadratic fields. Journal of Cryptology, 1, 107-118.

Chuang, I. L., Gershenfeld, N., & Kubinec, M. (1998). Experimental implementation of fast quantum searching. Physical review letters, 80(15), 3408.

Diffie, W., & Hellman, M. E. (1976). New Directions in Cryptography. IEEE Transactions on Information Theory, 22(6), pp. 644-654.

Hallgren, S. (2007). Polynomial-time quantum algorithms for Pell's equation and the principal ideal problem. Journal of the ACM (JACM), 54(1), 1-19.

Hoffstein, J., Pipher, J., & Silverman, J. H. (2006). NTRU: A ring-based public key cryptosystem. In Algorithmic Number Theory: Third International Symposium, ANTS-III Portland, Oregon, USA, June 21–25, 1998 Proceedings (pp. 267-288). Berlin, Heidelberg: Springer Berlin Heidelberg.

Ishai, Y., Prabhakaran, M., & Sahai, A. (2008). Founding cryptography on oblivious transfer–efficiently. Advances in Cryptology–CRYPTO 2008: 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings 28 (pp. 572-591). Springer Berlin Heidelberg.

Koblitz, N. (1987). Elliptic curve cryptosystems. Mathematics of computation, 48(177), 203-209.

Kitaev, A. Y. (1997). Quantum computations: algorithms and error correction. Russian Mathematical Surveys, 52(6), 1191.

Kumar, A., & Garhwal, S. (2021). State-of-the-art survey of quantum cryptography. Archives of Computational Methods in Engineering, 28, 3831-3868.

Merkle, R. C. (1979). Secrecy, authentication, and public key systems. Stanford university.

Miller, V. S. (1986). Use of elliptic curves in cryptography (pp. 417-426). Springer Berlin Heidelberg.

Peikert, C. (2016). A decade of lattice cryptography. Foundations and Trends® in Theoretical Computer Science, 10(4), 283-424.

Perlner, R. A., & Cooper, D. A. (2009). Quantum resistant public key cryptography: a survey. In Proceedings of the 8th Symposium on Identity and Trust on the Internet (pp. 85-93).

PRNewswire. (2022). IBM unveils 400 qubit-plus quantum processor and next-generation IBM Quantum System Two. IBM Newsroom. https://newsroom.ibm.com/2022-11-09-IBM-Unveils-400-Qubit-Plus-Quantum-Processor-and-Next-Generation-IBM-Quantum-System-Two.

Rabin, M. O. (1981) How to exchange secrets with oblivious transfer. Technical Report TR-81, Aiken Computation Laboratory, Harvard University.

Regev, O. (2009). On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM), 56(6), 1-40.

Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2), 120-126.

Shor, P. W. (1999). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM review, 41(2), 303-332.

Thaler, J. (2022). Proofs, arguments, and zero-knowledge. Foundations and Trends® in Privacy and Security, 4(2–4), 117-660.

Wiesner, S. (1983). Conjugate coding. ACM Sigact News, 15(1), 78-88.