# An Introduction To The Topic Of Post-Quantum Cryptography

**Jack Harley**
School of Computer Science and Statistics
Trinity College Dublin
The University of Dublin
Email: jharley@tcd.ie
Student No.: 16317123

***Abstract*—Quantum computing promises to bring us many new techniques for solving hard problems, but we depend on the hardness of some of these problems to secure our data in the modern world. We will discuss (a) the dependency society has developed on modern cryptosystems and (b) the most prevalent cryptosystems currently in use today. We will analyse their vulnerability to the advancement of quantum computing; introduce some quantum-safe cryptosystems currently released and/or under development; briefly discuss standardisation efforts from NIST and predict future adoption trends. We conclude that there are many viable options available for post-quantum cryptography and that adoption may be a challenge for legacy devices.**

## I. Introduction

On the 24th of July 2018, Google released Chrome 68. There were a number of new features in the release including improvements for payments and handling page lifecycles, but the most important change was a small, simple UI change that signified a shift to a new era. We are all accustomed whether consciously or subconsciously to the green padlock icon at the top of our browsers which signifies our connection to the web server was conducted securely with HTTPS/TLS, with little to no possibility that an eavesdropper could intercept any of our communications. Tech-savvy users may even have consciously checked for that reassuring padlock before entering their credit card details on a website. But TLS is no longer a luxury, from Chrome 68 onwards it has been expected: any website which uses an unencrypted plain HTTP connection is marked with a ”Not secure” notice. Firefox has gone further in recent times, with all plain HTTP websites marked with a distressing red padlock with a line struck through it to warn the user: ”be careful!”

The modern connected world depends on cryptography to succeed. In 2020 with COVID-19 we have seen how many industries were able to transition their entire workforce online in an extremely short period of time. Would this be possible without encrypted video calls and email? Certainly not, many industries would be fearful of interception of their confidential information. Yet many take this hugely important facet of modern digital life for granted.

But this could all come crashing down with the advent of quantum computing. Quantum computers exploit quantum-mechanical phenomena including entanglement and superposition to perform computation. There's a lot to unpack in that sentence, and to fully explain the nature and workings of a quantum computer would take several hundred pages, but for the purposes of this report we can think of a quantum computer as a magic box that can solve certain difficult mathematical problems very fast. There are a number of problems with building a quantum computer. In regular computing the atomic element of state is the bit, which can hold a value of 0 or 1, and all of our computation can be boiled down to performing operations on those 0s and 1s. Quantum computing has an analogous element of state: the qubit. It turns out that building a quantum computer with a large number of qubits is incredibly difficult as they suffer from a quantum phenomenon known as ”decoherence” where the state of the qubit is lost due to interference from the environment or the measurements/operations we attempt to perform on it.

There are algorithms available for quantum computers that allow us to break the current widely used RSA/ECC asymmetric (public-key) cryptosystems. Breaking RSA-2048 for example would require 4099 completely stable qubits. As of 2020 the leading research quantum computers (that we know about) have somewhere between

50 and 80 qubits, and those small number of qubits are not stable and suffer from errors, so thankfully we have some time to prepare.

In the remainder of this report, we will first discuss the quantum attack vectors that current popular cryptosystems are vulnerable to. We will then move on to some possible cryptosystems and cryptographic techniques which are currently under research or available and are resistant to quantum attacks.

## II. Symmetric Encryption

Symmetric encryption uses the same key to both encrypt and decrypt data. It's widely used across technology of all types in the modern world. All iPhones and other devices based on Apple's mobile OS which they currently call iOS since the iPhone 3GS fully encrypt the device with AES-256 symmetric encryption. Popular disk encryption tools for traditional personal computers (PCs) such as Microsoft's BitLocker, TrueCrypt/VeraCrypt and LUKS/dm-crypt all utilise symmetric encryption. In addition to the most commonly used AES, other popular symmetric cryptosystems include ChaCha20, Serpent and Twofish. Notably, ChaCha20 is used by the recently released WireGuard VPN protocol, which will likely supersede the currently popular OpenVPN/IPsec over the next couple of decades.

The best currently known quantum attack against symmetric cryptosystems is through the use of Grover's algorithm. Grover's algorithm is a quantum algorithm which provides quadratic speedup for brute-forcing symmetric encryption. Essentially it allows finding an AES-128 key in the time it would take a classical computer to find an AES-64 key, or an AES-256 key in the time it would take a classical computer to find an AES-128 key. This is a reduction in security but in no way does it render current symmetric cryptosystems obsolete. The current recommendation for quantum resistance is to simply double the key size you are currently comfortable with in the context of classical brute forcing. If AES-128 is safe enough for you in the context of advancements in classical computing, AES-256 gives a similar level of security taking quantum computing into account.

If key sizes beyond 256 bits are desired, AES in its current form is not a viable choice since it is only defined for key lengths of 128, 192 and 256 bits. One possible option if very large key sizes were desirable is Kalyna. Kalyna is a cipher adopted as the national encryption standard of Ukraine in 2015. It is based on the design of AES with a number of improvements, and supports key sizes of 128, 256 and 512 bit keys. It should be noted that 512 bit key sizes are complete overkill and not necessary except for niche cases where data must remain secure for 100+ years.

## III. Asymmetric Encryption

Asymmetric encryption (also referred to as public-key encryption) uses different keys for encryption and decryption. Asymmetric encryption is incredibly important in the context of securing communications channels. If you wanted to connect to google.com, using symmetric encryption would require you to go to Google HQ and securely upload a secret key to their server which would then be used by you and Google to encrypt your communications. The key could not be sent digitally because there is a chance it would be tampered with by an adversary (a man in the middle or MITM attack). Asymmetric encryption is much more practical, it allows both parties to generate a pair of two keys: a public key and a private key. The parties then exchange their public keys. Each party can then encrypt data with the other party's public key, transmit it, and the receiving party can decrypt it with the matching private key.

There are a few different popular techniques for implementing asymmetric encryption. Two modern and highly popular asymmetric cryptosystems are RSA and ECDH. RSA is based on a problem in mathematics/computer science called the integer factorisation problem. Essentially if you take two large prime numbers $p$ and $q$ it is easy to multiply them together to get $N$, however if you start with $N$, it is incredibly difficult to find the original $p$ and $q$. ECDH (Elliptic-Curve Diffie-Hellman) is a way of securely negotiating a key with another party over an insecure transmission medium. It is based on a problem called the elliptic-curve discrete logarithm problem. Very simplistically, in a graphical sense, it involves an elliptic-curve which you bounce a point around using tangents and lines conforming to certain conditions. The difficult part for an attacker is figuring out how many of those "bounces" were executed. Please note that this is an extremely simplistic definition of ECDH, further reading is advised if you are interested in elliptic-curve concepts.

The important thing to takeaway from this is that almost all asymmetric cryptosystems in widespread use in 2020 depend on the difficulty of solving either the integer factorisation problem or discrete logarithm problem. Both of these problems are easily solved on a quantum computer using an algorithm called Shor's algorithm. Shor's algorithm allows for finding the prime factors $p$ and $q$ of $N$ in a short amount of time. With some modifications it can also be used to solve the

discrete logarithm problem. And so we have our major apocalyptic problem with quantum computing on the horizon: none of the current widespread public key cryptosystems are secure if an attacker has access to a quantum computer. We must investigate asymmetric cryptosystems which do not rely on problems that are easily solved by a quantum computer:

### A. Lattice-based Cryptography

Lattices are an interesting type of mathematical structure with many potentially quantum resistant cryptosystems designed on lattice-related problems. Graphically, a simple lattice can be thought of as a set of 2 basis vectors in 2D space. For example, $b\vec{1} = 2\vec{i} + \vec{j}$ and $b\vec{2} = \vec{i} + 3\vec{j}$. A lattice is all of the vectors produced by adding those vectors together different numbers of times. For example $b\vec{1} + b\vec{2}$, $2b\vec{1} + b\vec{2}$, $2b\vec{1} + 2b\vec{2}$, $b\vec{1} + 2b\vec{2}$, ...

There are a number of hard to solve problems relating to lattices. We will examine a few which are used in cryptographic applications. It is important to note that a 2D lattice such as the one given in the above example would have easy solutions to the problems described below. In order to gain hardness we must increase the dimensionality. The example is given only as an aid for understanding lattices, in the same way that the prime numbers 5 and 7 would not be suitable as primes for RSA encryption, but are easier to work with than numbers with 25 digits.

Here are some of the important computationally hard lattice problems:

- **The shortest vector problem (SVP)**: This involves finding the shortest vector possible in a lattice when given the basis vectors. Essentially if you have two basis vectors in $\vec{i}$ and $\vec{j}$, what is the closest vector to $0\vec{i} + 0\vec{j}$. This is a known hard problem.
- **The closest vector problem (CVP)**: This is a generalisation of the shortest vector problem. Given a lattice by two basis vectors in $\vec{i}$ and $\vec{j}$, and a vector $\vec{A} = a\vec{i} + b\vec{j}$, what is the closest vector to $\vec{A}$.
- **The shortest independent vector problem (SIVP)**: This is an extension of the shortest vector problem. Instead of finding just the shortest vector, you must now find the $n$ shortest vectors, and they must be linearly independent.

We will now discuss some implementations of cryptosystems based on lattices, their advantages, disadvantages and maturity:

*1) **NTRU**:* The first version of NTRU was developed in 1996 by Jeffrey Hoffstein, Jill Pipher and Joseph H. Silverman. It is based on the previously discussed shortest vector problem and has remained mostly secure despite 20 years of scrutiny by researchers. It has very fast operation, and small key sizes (smaller key sizes are very desirable in cryptography). There are a few possible attacks against NTRU when the parameters are not carefully chosen but the latest implementations of NTRUEncrypt (the publicly available program used to perform encryption and decryption) has functionality to ensure selection of secure parameters.

NTRU was originally under patent but was placed into the public domain in 2017, removing concerns from many that it was not suitable for use by the open source community due to potential legal issues.

NTRU Prime is a variant of NTRU which was released by Daniel Bernstein, Tanja Lange, Chitchanok Chuengsatiansup and Christine van Vredendaal in 2016. It alters some of the mathematical structures from the original NTRU which the developers were concerned may contain an attack vector. Despite their concern, to date no attack has been found against the algebraic structures they altered so there is no real reason to consider NTRU Prime an upgrade to NTRU.

*2) Learning with Errors/Ring LWE:* In 2005, Oded Regev released a paper on a problem he called the "Learning with Errors" problem. In his paper, he showed that the problem he described was as hard to solve as the previously discussed difficult lattice problems. The LWE problem from the perspective of an attacker can simplistically be explained as follows: there is a function $y = f(x)$, and you are given some values $y_i = f(x_i)$ for this function, some of which may be errors (hence "with errors"), what is the function?

If we take learning with errors and specialise it to operate only on polynomial rings over finite fields, we now have the "Ring Learning with Errors" problem, which is the basis for modern LWE cryptosystems. In 2014, Peikert described a cryptosystem for using this problem for key exchange, essentially he proposed a quantum-safe replacement for the currently in use ECDH/DH based on RLWE. This cryptosystem is commonly referred to as Peikert RLWE-KEX.

In 2015, several researchers improved on his work with a cryptosystem they called "NewHope", an improvement on RLWE-KEX. A C implementation of NewHope is available on GitHub and no attacks have been found against it since its release. Speed and key size make NewHope and RLWE-based cryptosystems in general, excellent candidates for the post-quantum era.

### B. Code-based Cryptography: The McEliece Cryptosystem

When a data storage or transmission medium is not 100% reliable, certain computing applications demand a guarantee that the data is free from errors due to potential catastrophic issues if such an error did occur. An example could be the guidance systems for an airplane or rocket reading the predetermined mass of the vehicle from memory to use for trajectory calculations. This would be particularly important for a vehicle operating in space since errors can be introduced by radiation. An error in this value could have disastrous outcomes including loss of life. Error correcting codes (ECC) resolve this issue, they encode additional redundant data with the original data, allowing errors not only to be detected but also corrected in real-time.

The McEliece cryptosystem exploits error correcting codes to perform cryptography. Specifically, McEliece uses Goppa codes, and depends on the fact that it is difficult to decode a general code without knowledge of its parameters.

McEliece is very interesting not because of complex mathematics behind it or because of any particular advantages the cryptosystems functionality itself conveys, but instead because of its maturity. It was first described in 1978, just one year after RSA and no serious attack vectors have been found against the algorithm since its inception. McEliece predicted that the parameters he used in 1978 to encrypt data would be broken with roughly $2^64$ computational cycles, and in 2008 this was indeed proven to be the case, with a parallel information set decoding algorithm breaking his original parameters in $2^60$ iterations. By increasing the parameter values (and therefore key sizes) McEliece can be made just as safe as AES-256 for the foreseeable future.

McEliece also has the advantage of being a fast algorithm, in fact it is much faster than RSA. The big downside is key size. For modern use and to maintain security against quantum computing speed up, a total key size of roughly 8 megabytes is required. This is huge compared to RSA (256 bytes for RSA-2048) which is already considered to have a fairly large key size compared to elliptic-curve based cryptosystems. This is a potential issue for low-speed connections as well as embedded and other low power devices where memory may be at a premium. It would be impractical to demand an 8MB key exchange on every single HTTPS connection for example, so some form of key reuse would be necessary, which is thankfully not a security issue for McEliece, but this does bring other challenges into play regarding cache times and storage requirements for both clients and servers that are currently not a problem with our current cryptosystems that utilise small keys.

It would be foolish to write-off McEliece because of the key size issue, it is unmatched in terms of the time it has stood against cryptanalysis. Those who are truly concerned that the NSA has a secret quantum computer with thousands of qubits could seriously consider adopting McEliece immediately, without real concern that unpublished, currently unknown attack vectors exist.

## IV. Standardisation Efforts

The National Institute of Standards and Technology (NIST) in the USA has been working on a process to select post-quantum cryptosystems over the last few years. NIST was responsible for the standardisation of AES in the year 2000, and their influence in the international cryptography community makes it likely that their standardisation effort will be the most successful.

In January 2017, NIST called for submissions of potential cryptosystems. Submissions were closed in November 2017 and the list of first round candidates was published in December 2017. Over the subsequent 12 months, over 10 attacks were published on various round one candidates and those candidates proven insecure were removed from the selection process. The round two candidates were published in January 2019. They include NTRU, NTRU Prime, NewHope and McEliece (all of which we discussed in this report) as well as a number of other candidates. Comments were accepted on the round two candidates but no new attacks were published. NIST has published a timeline in which they expect to conduct a third and final round this year (2020) with the aim to selecting final cryptosystems and publishing draft standards sometime between 2022-24.

## V. Future Adoption

Adoption will likely become possible after NIST publishes a final standard, sometime around 2025. Adoption on the web could be achieved quite quickly with a new HTTPS/TLS version incorporating a quantum-safe cryptosystem. Once support is added to the largest players in the browser industry (Google Chrome, Mozilla Firefox, Edge) and also the major players in the web server space (nginx, Apache, etc.) as well as the major reverse proxy providers such as Cloudflare, adoption would occur for users transparently without issue.

Adoption will also likely drive further research interest and improvements in security since there will be a significantly greater interest from governments and hackers

to exploit a cryptosystem that is actually in use in the wild.

One major issue will be legacy software and devices. As an example, legacy applications built on Java 6 and earlier versions do not support Diffie-Hellman exchange with a key size above 1024 bits, which is considered to be insecure for the past few years. These applications will in some cases persist for many years or decades before the software is eventually replaced. Similar issues will occur with post-quantum crypto adoption, which is why it is crucial that adoption occurs well before a quantum computer sufficiently advanced to break current cryptosystems is operational. A failure to adopt early will result in many additional years where insecure devices have to remain operational.

## VI. Conclusions

The integer factorisation and discrete logarithm apocalypse is coming, but it's coming slowly and the future is bright. Efforts from researchers have produced a number of viable crytosystems for the quantum era. The main issue with moving to quantum-safe cryptosystems will likely be legacy devices and inertia, which is par for the course in the technology world. The security implications and regulatory compliance requirements for those left behind will hopefully force the usually slower moving industries to adopt the new technology before vulnerability becomes a real concern.

## References


[1] Google. 2018. A milestone for Chrome security: marking HTTP as "not secure". https://www.blog.google/products/chrome/milestone-chrome-security-marking-http-not-secure/

[2] Andreas Baumhof. 2019. Breaking RSA Encryption – an Update on the State-of-the-Art. https://www.quintessencelabs.com/blog/breaking-rsa-encryption-update-state-art/

[3] EU PQCRYPTO. 2015. Initial recommendations of long-term secure post-quantum systems. http://pqcrypto.eu.org/docs/initial-recommendations.pdf

[4] Daniel J. Bernstein, Tanja Lange. 2017 Post-quantum cryptography: dealing with the fallout of physics success. http://eprint.iacr.org/2017/314

[5] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. 1998. NTRU: A Ring-Based Public Key Cryptosystem. In Proceedings of the Third International Symposium on Algorithmic Number Theory (ANTS-III). Springer-Verlag, Berlin, Heidelberg, 267–288. https://dl.acm.org/doi/10.5555/648184.749737

[6] NTRU PKCS Tutorial (with Recommended Parameters and Usage). https://web.archive.org/web/20120606210107/http://www.securityinnovation.com/security-lab/crypto/155.html

[7] Oded Regev. 2005. 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). Association for Computing Machinery, New York, NY, USA, 84–93. DOI: https://doi.org/10.1145/1060590.1060603

[8] Chris Peikert. 2014. Lattice Cryptography for the Internet (RLWE). https://eprint.iacr.org/2014/070.pdf

[9] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, Peter Schwabe. 2015. Post-quantum key exchange – a new hope. https://eprint.iacr.org/2015/1092.pdf

[10] R. J. McEliece. 1978. A Public Key Cryptosystem Based On Algebraic Coding Theory. https://ipnpr.jpl.nasa.gov/progress_report2/42-44/44N.PDF

[11] Suanne Au, Christina Eubanks-Turner, Jennifer Everson. 2003. The McEliece Cryptosystem. http://www.math.unl.edu/~s-jeverso2/McElieceProject.pdf

[12] NIST Post-Quantum Cryptography Project. since 2016. https://csrc.nist.gov/Projects/post-quantum-cryptography

[13] Miklós Ajtai and Cynthia Dwork. 1997. A public-key cryptosystem with worst-case/average-case equivalence. In Proceedings of the twenty-ninth annual ACM symposium on Theory of computing (STOC '97). Association for Computing Machinery, New York, NY, USA, 284–293. DOI: https://doi.org/10.1145/258533.258604