CS3031: Post-quantum Cryptography

Conor McCauley, 17323203

***Abstract*—Cryptography is an essential aspect of modern computing and telecommunications as it facilitates secure and private communication. In this paper we will discuss the vulnerabilities many standard cryptographic algorithms have when exposed to quantum computing as well as some of the cryptographic approaches that are protected against quantum attacks.**

I. Introduction

Cryptographic methods are essential in modern computing and telecommunications. Cryptographic methods and algorithms provide a means to facilitate secure and private communication and data transfer between individuals while in the presence of potentially adversarial third parties. Most modern cryptographic algorithms derive their strength and security from the fact that they are computationally infeasible to break. For example, many public-key cryptographic algorithms, such as RSA, make use of very large prime numbers when generating encryption and decryption keys due to the fact that it takes an inordinate amount of computational time to calculate the prime factors of very large numbers even with the most efficient known methods.

However, there do exist algorithms and methods, such as Shor's algorithm, that would reduce the computational time and power needed to factorise large numbers by very drastic amounts when they are run using quantum computers. If such algorithms were to be successfully implemented the resulting improvement in computational efficiency would completely compromise the security of a significant portion of global telecommunications. It is therefore necessary for 'quantum-proof' cryptographic algorithms to be developed and deployed before such a disaster can occur.

II. Vulnerabilities of Public-key Cryptography

Public-key cryptographic algorithms are widely used in modern computing. One of the most prominent examples of public-key cryptography would be the combined use of Diffie-Hellman key exchange and RSA in the Transport Layer Security (TLS) protocol which allows for secure communication between nodes on a computer network. This communication can be in many forms including web browsing, instant messaging and email. The TLS protocol encrypts the data that is transferred between servers and their clients which prevents adversarial third parties, including the public, from decrypting and reading the data. This security is clearly of paramount importance in the digital age when we consider the prevalence and necessity of online bank transfers, user authentication, digital signatures and encrypted messaging clients. Further examples of public-key cryptography are its use in blockchain technologies such as Bitcoin as well as its use in PGP.

As previously mentioned, most public-key cryptographic methods derive their strength from the inability of classical computers to factorise large numbers efficiently. In the RSA system, for example, public and private keys are generated, roughly speaking, through the multiplication of two very large prime numbers. Provided these prime numbers are large enough, i.e., 1024 bits, the most efficient classical algorithm that we know of would still require an astronomically large amount of computing time to factorise the product of the prime numbers and consequently decrypt any susceptible data.

Unfortunately, cryptographic algorithms of this kind are vulnerable to quantum computing attacks via Shor's algorithm. Shor's algorithm utilises the mathematical fact that any two numbers, N and a, can be related through the following equation: $a^b = m \cdot N + 1$, for some numbers b, m. In this case, N could be our product of two very large primes. Determining the value of b would, in many cases, allow for the subsequent factorisation of N. Shor's algorithm, when implemented on a classical computer, is very inefficient. However, when implemented on a quantum computer, it can take advantage of the fact that the quantum computation can be set up such that all possible guesses for the value of b are checked simultaneously but, through destructive interference, only the correct guess is output at the end.

Fortunately, Shor's algorithm has only been used in real-world quantum computers to factor numbers as large as 21 due to the extreme difficulty in running programs on quantum computers of any significant size [1]. However, given the current speed of scientific research and discovery in fields such as quantum computing it would certainly appear prudent to immediately begin developing and implementing quantum-proof cryptographic methods.

III. Symmetric-key Cryptography

Symmetric-key cryptography, in which only a single private key is used to both encrypt and decrypt messages and data, is also extremely common. It is used in the Advanced Encryption Standard (AES), Blowfish and RC4, to name just a few of its applications. Symmetric-key cryptography is much faster than public-key cryptography and, as such, it sees frequent use in the encryption and decryption of large volumes of data, e.g., databases or files.

Unlike public-key cryptography, symmetric-key cryptography already appears to be fairly resistant to quantum computing attacks. While some quantum algorithms, such as Grover's algorithm, are capable of reducing the amount of computation needed to break symmetric-key encryption their effectiveness is limited due to the fact that a simple doubling of the key sizes used render the quantum attacks useless.

IV. Lattice-based Cryptography

Lattice-based cryptography involves the development of cryptographic protocols that make use of computationally

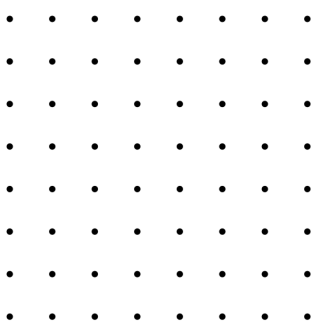

Fig. 1. A square lattice in the Euclidean plane

expensive problems based on lattices.

A. Definition of a Lattice

A relatively simple definition of an n-dimensional lattice can be given as follows: given a set of vectors such that each element in each vector is a member of the group $\mathbb{R}^n$, a lattice is formed by the subgroup of all the linear combinations of the vectors and any integer coefficients. For example, given the vectors $[1\ 0]$ and $[0\ 1]$ the resulting lattice would simply consist of every point on the Euclidean plane with integer coordinates (fig. 1). Lattices are defined by organising their basis vectors into a matrix. As such, the example of a lattice that we just gave could be written out like so:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

B. Shortest Vector Problem

One very simple example of a computationally expensive lattice-based problem is the shortest vector problem (SVP). Given some lattice, we are asked to find the shortest non-zero vector contained within it, i.e., find the shortest non-zero distance between two lattice points. In the two-dimensional example we gave in the previous section the answer would simply be 1. However, for higher dimensional lattices with a large number of basis vectors the SVP takes an exponential amount of time to solve.

C. Lattice-based Public-key Encryption

In this section we will briefly describe a computationally hard method of public-key encryption using lattices.

1) Key generation: To generate a public-key we take some random lattice matrix, L, and some integer, q, where each element in L is taken modulo q - we also make both L and q public. We then generate some private vector, $\vec{v}$, consisting solely of ones and zeroes and multiply that by L to produce our public key $\vec{u} = L\vec{v}$. The vector, $\vec{v}$, will act as our private key.

2) Message encryption: Another user can send us an encrypted message by first generating three random private vectors $\vec{s}$, $\vec{t_1}$ and $\vec{t_2}$. The user will then use those vectors, as well as our public matrix, L, to calculate the following public vector: $\vec{p_1} = L\vec{s} + \vec{t_1}$. Finally, for each bit, b, in their message they will encrypt it via the following equation using our aforementioned public key: $p_2 = \vec{s}\vec{u} + \vec{t_2} + \frac{bq}{2}$.

3) Message decryption: In order for us to decrypt each bit in the message we simply multiply $\vec{p_1}$ by our private key, $\vec{v}$, and subtract this from p_2. If the resulting value is very close to zero then we know that the bit in question was also a zero. If the result is not close to zero then we know that the bit was a one. We can repeat this process for each bit in the message.

D. Resistance to Quantum Attacks

The previously described method is proven to be as computationally hard as the problem know as 'learning with errors' which is believed to be secure against quantum attacks [2]. Therefore, lattice-based public-key encryption provides a potentially quantum-proof solution to public-key encryption.

E. Advantages of Lattice-based Cryptographic Systems

Evidently, the primary advantage of lattice-based cryptographic systems are their resistance to quantum computer attacks. Another advantage, which will also be discussed in the next section, is the ability of lattice operations to be parallelised due to their being represented via matrices.

F. Disadvantages of Lattice-based Cryptographic Systems

A significant disadvantage of lattice-based cryptography is that, due to the way lattices are represented by matrices, cryptographic algorithms involving lattices must rely heavily on matrix multiplication and similar operations. These types of operations are computationally expensive and involve a significant amount of memory. However, as previously mentioned, the potential parallelisation of matrix operation also pose a prospective advantage.

V. Multivariate Cryptography

Multivariate cryptography involve the use of multivariate polynomials (usually in quadratic form) as public keys for asymmetric cryptographic algorithms.

A. Definition of a Multivariate Quadratic Polynomial

A multivariate quadratic polynomial is simply, as the name suggests, a quadratic polynomial that contains multiple variables. For example, the following equation is a quadratic polynomial containing two variables, x_1 and x_2, and some coefficients, $c_1, \cdots, c_6$:

$$c_1x_1^2 + c_2x_2^2 + c_3x_1x_2 + c_4x_1 + c_5x_2 + c_6 = 0$$

We can express a multivariate quadratic polynomial containing m variables, $p(x_1, \cdots, x_m)$, like so:

$$p(x_1, \cdots, x_m) = \sum_i a_i x_i + \sum_i b_i x_i^2 + \sum_{i>j} c_{ij} x_i x_j$$

where a_i, b_i and c_{ij} are the coefficients in the equation.

B. Using Multivariate Polynomials as One-way Functions

Given a known multivariate polynomial, p, containing m variables we can easily determine the output, y, of this polynomial for some given variable values, i.e., the polynomial $p(x_1, x_2) = x_1x_2 + x_1 + 2$ evaluates to 6 for the input $p(2, 1)$. However, given some output, y, it is much more computationally difficult to determine the variables, $x_1, \cdots, x_m$, that would result in that output. In fact, there is no known polynomial-time algorithm that allows us to solve this problem on neither classical nor quantum computers. This computational hardness allows us to effectively use large multivariate polynomials as one-way functions [3].

C. Asymmetric Encryption with Multivariate Polynomials

Using the aforementioned property of multivariate polynomials it is possible to implement an asymmetric cryptographic system. We will briefly describe a simplified version of one possible system in this section.

1) Key generation: We generate some random series of n multivariate polynomials $\mathcal{P} = (p_1(x_1, \cdots, x_m), \cdots, p_n(x_1, \cdots, x_m))$ and some random easily invertible map, $\mathcal{T}$. We can then compute $\mathcal{P} \circ \mathcal{T}$ which will act as our public key while our private key will be $\mathcal{T}$.

2) Message encryption and decryption: To encrypt a message we, in simplified terms, pass each block of the message through our public map $\mathcal{P}$. Conversely, to decrypt a message we can pass each block through our private map $\mathcal{T}$.

D. Security of Multivariate Cryptography

While the general hardness of solving multivariate systems of equations is known, there has not yet been a successful implementation of a secure cryptographic system based on multivariate polynomials. However, such protocols, if implemented, are believed to be resistant to attacks from quantum computers.

E. Advantages of Multivariate Cryptographic Systems

As with lattice-based cryptography, there is no known algorithm that gives quantum computers a significant advantage over classical computers in breaking multivariate cryptographic systems. Another property of multivariate polynomials is that the mathematical operations that are involved in implementing the cryptographic algorithms consist solely of addition and multiplication and are carried out on fields of relatively small numbers. This property means that efficient multivariate algorithms can be implemented that use a small amount of resources and can be run on hardware such as integrated circuit cards.

F. Disadvantages of Multivariate Cryptographic Systems

One of the previously mentioned disadvantages of multivariate cryptographic systems is that no implementation, thus far, has been found to be secure. However, as such systems are relatively new it is not unlikely that more and more secure systems will eventually be discovered. Another disadvantage of multivariate systems is their reliance on keys that are very long. Some implementations, such as the unbalanced oil and vinegar scheme, involve public keys that are many kilobytes in length.

VI. Other Post-quantum Cryptographic Methods

A. Hash-based Cryptography

Hash-based cryptography involves the use of hashing functions to digitally sign messages. Hash-based digital signatures already see popular use in blockchain technologies such as cryptocurrencies like Bitcoin. However, the common method that is used to generate digital signature, the Elliptic Curve Digital Signature Algorithm (ECDSA), is vulnerable to quantum computers running Shor's algorithm.

Fortunately, there are methods that are proven to be resistant to quantum attacks. One such method, known as the Merkle signature scheme, involves the use of structures known as Merkle trees to combine a number of hashing keys that, on their own, could only be used to each sign a single message once, so that multiple messages can be signed using the same key structure.

The Merkle signature scheme's security derives from its use of hash functions as opposed to the vulnerable methods of integer factorisation and elliptic curves that are commonly used. Furthermore, the scheme's ability to sign multiple messages using the same key structure allows it to be used in public-key asymmetric cryptography.

A version of the Merkle signature scheme, eXtended Merkle Signature Scheme, has already been implemented in the Quantum Resistant Ledger, an open source blockchain platform that is resistant to quantum computer attacks.

B. Supersingular Isogeny Diffie–Hellman Key Exchange

Supersingular isogeny Diffie-Hellman key exchange (SIDH) is a cryptographic method that can be used to implement public-key asymmetric encryption that is based on the elliptic curve Diffie-Hellman (ECDH) method of key exchange. Like ECDH, SIDH derives its security from elliptic curves. SIDH, however, relies on a particular form of elliptic curves known as supersingular elliptic curves.

According to current research, SIDH is slightly less secure when attacked by a quantum computer than when attacked by a classical computer, but, it is still computationally hard enough to provide a suitable degree of security against quantum algorithms - making it a potential alternative to our current forms of elliptic curve key exchange.

VII. Conclusions

Although the current complexity and power of quantum computers is quite limited, it is not unreasonable to assume, given the fact that real-world quantum computers only came into existence in the late 1990s, that over the coming decades significant strides will be made in quantum computing that will result in legitimate threats to current cryptographic algorithms and methods.

As discussed in the beginning of this paper, it is clearly of the utmost importance that we maintain the security of global

telecommunications due to their ubiquity and vital role in the modern world. In order to do this, we must begin implementing algorithms that are protected against quantum attacks in the real-world by replacing the potentially vulnerable protocols that are currently in use.

However, given the sheer scale of global telecommunications and the computing industry in general, it is clear that such large and consequential changes to a key portion of information security protocol would require a significant degree of planning and oversight in order to make the transition safely.

We also briefly outlined some of the potential algorithms that could be used to replace our current methods if the aforementioned transition were to take place.

While each of the post-quantum algorithms that were mentioned have their advantages and disadvantages, they can each serve a purpose in the post-quantum cryptographic sphere. For example, multivariate cryptographic systems are very efficient and use only the most basic arithmetic operations and, as such, are suitable for low-resource devices and hardware - although no secure implementation is currently available. Hash-based cryptographic systems are already being widely used in blockchain technology and quantum-proof solutions are already in use. Based on my research, lattice-based cryptographic systems appear to be the most reliable and versatile solution to the threat posed by quantum computing. Finally, the SIDH method of public-key exchange provides a suitably similar replacement to the current, vulnerable ECDH method.

In summary, many viable cryptographic systems and solutions are available that are immune to quantum attacks, and, given the relative infancy of quantum computing, it would appear that there is more than enough time remaining to replace and secure our current global telecommunications systems before they can be compromised.

REFERENCES

[1] Martín-López, E., Laing, A., Lawson, T. *et al* (2012). Experimental realization of Shor's quantum factoring algorithm using qubit recycling. *Nature Photon* 6, pp. 773–776.

[2] Regev, Oded. (2005). On lattices, learning with errors, random linear codes, and cryptography. *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing - STOC '05*, pp. 84–93.

[3] Ding J., Yang BY. (2009). Multivariate Public Key Cryptography. *Bernstein D.J., Buchmann J., Dahmen E. (eds) Post-Quantum Cryptography. Springer, Berlin, Heidelberg*, pp. 193-241.