Cryptography in the Post-Quantum Era

Rvail Naveed

***Abstract*—In recent years there has been a surge of interest and development in quantum computing. Quantum computers have the ability to solve complex mathematical problems that classical computers are not able to. This puts modern encryption systems at risk. This report aims to review the concept of ”post-quantum cryptography” and what it would look like in the age of large, scalable quantum computers.**

***Index Terms*—Quantum Computing, Cryptography, Public-Key Encryption**

I. Introduction

SINCE the rise of the internet in the 1990’s, there has been a very urgent need for crpytography. Cryptography, derived from the Greek word for ”hidden” and ”writing” has become a staple of every modern technology in use today. It is essential to countless businesses and the security, integrity and confidentiality of our ever-growing online presence as a society. Over the past few decades there have been significant strides made in the cryptography world (such as RSA and AES). Modern encryption algorithms rely on the hardness of solving one of two problems: *Integer Factorization* and the *Discrete Logarithm Problem*. These problems prove to be very difficult to solve modern classical computers as the power and time to solve them is immense. However, a relatively new phenomenon, Quantum-Computers, threaten this security.

Quantum computing began in the early 1980’s when a physicist named Paul Benioff proposed a quantum mechanical model of the Turing machine. Richard Feynman, a theoretical physicist later brought up the idea that a quantum computer may be able to perform tasks that a classical computer did not have the capacity for.

This paper aims to review Quantum Computers, how they operate, modern encryption algorithms and how a quantum computer threatens their security.

II. Quantum Computers

Quantum Computers can perform calculations based on the probabiltity an objects state before it’s measured. Classical computers encode data in bits. These bits can have *either* a state of 0 or 1. Quantum computers, on the other hand use *qubits* to perform their caclulations. Qubits are undefined properties of a system and have the ability to be in superposition. This means that a qubit can be in both states at the same time. Superposition can be thought of as a spinning coin, as the coin is spinning, it is neither a heads or a tails until it lands. Qubits are plugged into special algorithms that allows them to crunch through a vast number of outcomes simultaneously. To put this into perspective, 1 qubit can be in a superposition of 2 states, 3 qubits can be in a superposition of 8 states and 50 qubits can be in a superposition of 250 = 1,125,899,906,842,624 states at once. As of 2019 the most powerful quantum computer is owned by IBM with 53 qubits.

Another phenomenon that makes quantum computing so powerful is *Quantum Entanglement*. Entanglement allows for one or more qubits to become a pair such that a change in one qubit will directly result in a change in the other. THis happens, regardless of the distance netween them. Quantum Computers can harness the power of entangled qubits in a ”daisy chain” leading to exponential increases in computing power and speed.

However Quantum Computers are much more error prone than their classical counterparts due to *decoherence*. Qubits are very susceptible to errors from outside factors such as heat and vibrations. This causes their quantum behvaiour to decay and become unreliable. Qubits are very fragile and any noise can cause problems in their superposition. Despite best efforts, researchers have found it difficult to reduce errors due to noise.

There is extensive research being done on Quantum Computers and it won’t be long before quantum computers are at a point where they become useful. It is estimated that by 2030 a Quantum computer could be built for a budget of a billion dollars that could crack RSA-2048.

III. Current Cryptography

Current cryptography techniques can be divided into 2 distinct types: *Symmetric* and *Asymmetric*. These cryptography techniques rely heavily on the hardness of solving problems such as Integer Factorization and the Discrete Logarithm problem.

A. Symmetric

Symmetric-key cryptography algorithms rely on a single key to encrypt and decrypt messages. This key can be shared between two or more users. The plaintext message is entered into a ”cipher” that encrypts the message using the key and outputs it as ciphertext. In much the same way, the ciphertext is decrypted using the key and can be converted back to it’s plaintext form.

Symmetric Cryptography allows for fairly high security with fast speeds, however there is a significant risk of keys being intercepted while they are being dsitributed.

B. Asymmetric

Also known as public-key cryptography, it makes use of two keys compared to the one in symmetric. The pair of keys consists of a *public key* and a *private key*. The sender uses the receivers public key to encrypt a message before sending it. The receiver then decrypts the message using their own related private key. Public-key cryptography is essential to internet applications such as the Secure Socket Layer, Transport Layer Security and HTTPS.

Digital signatures can also be verified using public-key techniques. Digital signatures are a way of validating the integrity of data. A hash is of the data to be signed is generated, this hash is then encrypted using the private key. The hash combined with other metadata is then what's known as the digital signature. When the recipient recieves the message, it can be verified by running the message through the same hashing algorithm. A match indicated a valid, untampered message.

C. Integer Factorization Problem

RSA first proposed by Rivest, Shamir and Adleman in the 1970's, is one of the world's first public key cryptosystems. RSA is often used in combination with symmetric key algorithms for maximum security and speed. RSA encryption is designed to be easy to compute in one direction and difficult to compute in the opposite direction. Such functions are often called *trapdoor* or *one-way* functions. RSA relies on the integer factorization problem via prime numbers.
The operation of RSA is as follow:

- Two large primes are generated p, q such that they are far away from each other, otherwise it would be easier to crack.
- The product of the two primes is then found $n = p * q$ and $\phi = (p-1) * (q-1)$
- A random number $1 < e < \phi$ is selected such that the greatest common divisor (gcd) = $gcd(e, \phi) = 1$
- Compute a unique integer $1 < d < \phi$ such that $e * d = 1(mod\ \phi)$
- We then have (d, n) and (e, n) as the private and public key's respectively.
- Encrypt message m using $c = m^e * mod\ n$
- Decrypt: $m = c^d mod\ n$

D. Discrete Logarithm Problem

Given a finite cycle group Z^*p of order $p-1$ and a primitive element $\alpha \in Z^*p$ and another element $\beta \in Z^*p$, the DLP is determining an integer $1 \leq x \leq p-1$ such that $\alpha^x \equiv \beta modp$. x is called the discrete logarithm of β to the base α.

$$x = log_\alpha \beta modp$$

. The difficuly of solving DLP relies in finding an x that satisfies the equation. DLP can prove to be a very hard problem to solve, if the parameters are large.

E. Security

Both symmetric and public-key cryprography are considered to be very secure by today's standards. However this security is threatened as we move towards a reality where Quantum Computers are more powerful. The inherent security of symmetric cryptography is based on the length of the keys used for encryption/decryption. A 128 bit key length would take billions of years to crack on classical hardware. NIST recommends a move to 256 bit key lengths, which are thought to be theoreticallly resistant to Quantum attacks. The same cannot be said for public-key algorithms such as RSA and Elliptic Curve Crptography as quantum algorithms such as *Shor's Algorithm* and *Grover's Algorithm* aim to drastically reduce the amount of time it takes to crack them. With classical hardware, prime factorization is extremely time consuming and unimaginable, with the best time being $O(exp(\sqrt[n]{\frac{64}{9}n(logn)^2}))$. However, with Shor's algorithm this time is significantly reduced to $O(n^3 logn)$, where n the bits of the product $p * q$.

IV. Shor's Algorithm - This defo needs to be better

In 1994, Peter Shor developed a quantum polynomial-time quantum algorithm for integer factorization. This algorithm described a process for finding factors of a number, even a very large one in a very short amount of time. Integer Factorization serves as the basis for many of the worlds currently most secure crpytography algorithms, such as RSA. In Layman's terms, Shor's algorithm takes a guess at a number that could share factors with N (but most likely doesn't) and transforms that guess into a much better one that is very likely to be a number that shares factors with N. Understanding how algorithms like RSA operate is fundamental to understanding why Shor's Algorithm works. Shor's algorithm leverages the concept of quantum superposition mentioned earlier to compute factors of these very large numbers at a scale that is unmatched by modern classical computers. In order to compute factors of N

- Consider N, a very large number (eg: An RSA public key)
- In order to crack RSA, we neeed to find factors of N such that $f(x) = (x^r) modN$.
- Shor's Algorithm shifts the focus to finding period r i.e: $f(x) = f(x + r)$.
- The algorithm uses something known as a *Quantum Fourier Transform*(QFT).
- A quantum computer can be in many states simultaneously, which leads to very fast computing.
- This allows the computer to calculate the period r for all points simultaneously.
- Then the QFT essentially takes these values, transforms them into waves in such a way that they destructively interfere with each other, leaving only the correct value of r.
- This r can then directly be used to solve for a factor of N, thus breaking the encryption.

Shor's algorithm also applies to cryptography schemes that depend on the *Discrete Logarithm Problem*.

V. Grover's Algorithm

What it works on, how it operates

VI. Algorithms Threatened by the Quantum Era

VII. Quantum Key Distribution

Quantum Key distribution or QKD, first proposed in 1970, is a means of secure communication over a channel. This is achieved via a crytographic protocol based on the fundamentals of quantum mechanincs. QKD enables 2 or more people to share messages using a shared random key, much like traditional cryptpgraphy. In QKD , information is encoded in single photons of light which require special hardware to create and transmit. QKD's use lies in generating secure cryptographic keys for use in conjunction with other cryptograpghy schemes such as RSA, it would be too unreliable and inefficient at directly sending information. QKD in essence, works as follows; Two users Alice and Bob wish to communicate with each other but are unsure if someone is eavesdropping on theier conversation. Alice tries to send a message to Bob using a "bitstream" that is encoded in a base of her choosing. If Bob also chooses the same base to decode the message, then their results correlate, otherwise they are random and are discarded. If someone tried to intercept the communication, they would introduce errors into the bitstream, this is discussed further below. This approach to key distribution comes with a number of advantages, namely that a secret key can be created in a provably secure way to prevent any any eavesdropping of private information, eavesdropping is easily detectable and perhaps most importantly, QKD has the ability to wasily integrate with current cryptography to provide additional security. In this scenario, the interceptor would not only have to break the quantum key but also the classical one.

There are two main types of QKD protocols: *Prepare and Measure* and *Entanglement Based*.

A. Prepare and Measure

Measurement one of the fundamental concepts of quantum mechanics. Measurement of a quantum state changes that state in some way. This fact of quantum mechanics can be taken advantage of to detect eavesdropping, as the very act of eavesdropping requires some level of measurement. Prepare and Measure techniques also allow us to see *how much* information has been intercepted. The most common prepare and measure protocol is BB84 which is discussed further below.

B. Entanglement Based

Entanglement based protocols exploit the quantum concept of entanglement which was explained earlier. Two or more qubits states become dependant on each other and they are in a joint state of superposition. Trying to determine the state of one of those qubits will directly result in a change in the other, making it very easy to detect eavesdropping. Theoretically, entanglement based protocols are much more secure than their prepare and measure counterparts, however it is currently unfeasible as managing and creating entangled qubits is very difficult.

C. BB84 Protocol

BB84 was the worlds first Quantum Key Distribution protocol. It was created in 1984 by Bennet and Brassard. BB84's operation can be divided into three layers:

1) Physical Layer:

This layer is the most hardware intensive portion of BB84. This is where communication between two parties, Alice and Bob, takes place. Alice chooses random photons to send to Bob, these can either be encoded with a 1 with 50% probability or a 0 with 50% probability. She can then encode the photons in either "base" X or Y, each with equal probability. These encoded photons are then sent over a quantum channel to Bob. Bob does not know what base Alice used or any information about how the qubits were encoded. He measures each bit on a random basis in either X or Y each with 50% probability. This leads to Bob having 75% of the correct bases for the qubits.

2) Key-Extraction:

In the next layer BB84 becomes classical to etract the key from the qubits that were sent by Alice. This layer consists of 4 sublevels. In the first sublevel called *sifting*, Alice and Bob reveal which bases they used and compare them over a public channel. They then proceed to discard the bits that do not correlate with each other.

In the 2nd sublevel *authentication*, Alice and Bob compare a subset of their bits to determine if their communication channel was compromised. If there is a higher error than can be accounted for by random noise, then the channel is believed to have been intercepted by a third party. This is because in order for an attacker to intercept the qubits, they must also guess the base wrong 50% of the time, leading them to forward some incorrect bits to Bob.

Error correction algorithms are then applied to reduce the effect of random noise on the data.

In the final sublayer *Privacy amplification*, the key generated from BB84 is combined with classical cryptography algorithms to reduce the effects of any undetected, minor eavesdropping and to maximize the security of QKD. This is done by encrypting the key itself using a scheme such as RSA or AES. This will require two levels of decryption.

3) Key-Application:

The key-application layer is then responsible for using this generated key to encode data.

D. E91 Protocol

The E91 protocol is similar in nature to BB84 escept it operates on entangled qubits. Alice and Bob both possess half of an entangled state. In this scenario there is nothing for an eavesdropper to intercept as the state of a qubit only settles after it has been measuered. As before Alice and Bob choose 1 iut of 2 bases to measure in, each with 50% probaility of getting chosen. After the qubits have been measured, they both share what bases they used over a public channel. If the qubits that were measured in different bases violate Bell inequalities, then states of the qubits remain entangled and it can be concluded that to eavesdropping has occured and the channel is secure. This is again, much more secure in theory than BB84 but it is impractical of the aforementioned decoherence problem of entangled qubits.

VIII. Post Quantum Crpytography

IX. Conclusion

The conclusion goes here.

Appendix A
Proof of the First Zonklar Equation

Appendix one text goes here.

Appendix B

Appendix two text goes here.

Acknowledgment

The authors would like to thank...

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.