Quantum Computing and its Application in Cryptography

Abstract. Quantum computing is the technology that allows complex operation and representation of computer state utilizing properties of photons such as superposition [6] and entanglement [13] as described by the theory of quantum mechanics. Therefore, quantum computing plays an increasingly important role in cryptography due to innovations it introduced such as quantum phase estimation and quantum key distribution [7]. These innovations can reinforce, or compromise tools and schemes used in classical cryptography e.g. Shor's algorithm [3] and one-time pad. In this paper, a simulation of prime factorization with Shor's algorithm is performed using qiskit. A success rate of 55% is achieved in 503 attempts. In response to the advancement of quantum computers and its ability to break popular encryption schemes as RSA, post-quantum classical cryptography protocols such as XMSS [4] and LMS [12] are introduced.

Keywords: Quantum Computing, Cryptography, Shor's Algorithm, Quantum Fourier Transform, Quantum Phase Estimation, Quantum Key Distribution

1 Introduction

Quantum computers are the next step in the innovation of conventional computers because they increase the speed and power and decrease the size of computers and communications [9]. Therefore, computer applications such as weather forecasting, predicting future events with multiple scenarios, and artificial intelligence are prime uses for quantum computing because it uses the principles of quantum bits to output more information and enhance the amount of processes that it can handle. Complex operations such as finding the prime numbers of a large number can now be tackled with relative ease. This paper focuses on the importance of cryptography, the study of writing or solving codes. Since information on a quantum computer is stored in photons, quantum computation can use photons' many states to create encryption for security keys as well as break encryption. This principle is the foundation of quantum computing's application in cryptography, providing a new world of computer security [11].

In this paper, section II summarizes the basic working principles of quantum computing, section III covers quantum computing's application in cryptography, including prime factorization, post-quantum and XMSS, section IV covers prime factorization with Shor's algorithm and a simulation of the algorithm is performed using qiskit to examine its efficacy, section V covers quantum key distribution, and the relevant innovations.

2 Fundamentals of Quantum Computing

The fundamentals of quantum computing rely on two main principles: interference and entanglement. These principles are used to determine the wave and particle-like aspects of photons, which are used to store information for encryption. Quantum computing consists of quantum bits and gates that provide tools for expressing complex algorithms in an abstract manner to execute on a quantum computer. In interference, the purpose is to provide a way to filter out all the infinitely possible answers to a solution from superposition and provide a single solution which is going to be explain further on the next section. In entanglement, the principle is to have multiple particles interacting with each other so that there is a correlation between their states, depending on what entanglement they have [1].

2.1 Quantum Bits

A quantum bit, or a qubit, is the fundamental basic unit of quantum information in computing. The qubit is usually compared to the classical bit in computers, but unlike the bits that carry information using 0 and 1, the qubit is a two-state quantum mechanical system that attempts to display the peculiar properties of quantum mechanics such as the spin or the polarization of a photon. Superposition occurs when a qubit is in a state between $|0\rangle$ and $|1\rangle$, with probability to collapse into either of them when measured. Quantum bits uses this principle of superposition, where the state of the qubit can be in both 0 and 1 at the same time. [1]

A quantum bit can be produced using the electrons of a specific atom. The electron has a magnetic dipole called a spin, where the magnetic field can determine the direction of the spin. When the spin is facing the direction of the magnetic field, the spin can be interpreted as a 1, and vice versa. To differentiate the spin of the electron, thermal energy can be used to change the dipole, and as the electron is switching from a spin down to a spin up state, from there, we are able to created a specific quantum superposition state, where the electron is somewhere in between the spin down and spin up state [11].

2.2 Quantum Gates

A quantum gate can be compared to classical logic gates because both are logic gates. Both contain a structure that takes a set inputs and expels a single output. However, a quantum gate has several properties that are unique to quantum mechanics. The first difference is that these gates rely on qubits to operate and can utilize superposition, entanglement, and reversibility. For reversibility, the quantum gate can be reversed, meaning that these gates will never lose information.

Mathematically, the quantum gates are composed of matrices and vectors. In "braket" notation often used when describing states of quantum computers, state vector ψ is displayed in 'bra' $\langle\psi|$ as row vector and 'ket' $|\psi\rangle$ as column vector, the values inside the bras and kets represents the values in the vector. For example, a column vector denoting a quantum state analogous to classical bit state 0 are often presented as $|0\rangle$. An often-used quantum gate is the Hadamard gate. This gate changes the input qubit from a computational basis of 0's and 1's into a state of superposition. This gate is one of the more important quantum gates, as it allows transformation of classical states into superposition. Without it, a quantum computer would not be any different than a classical computer. There are also other gates for numerous applications, such as the Pauli gates, Toffoli gates, Deutsch gates, and controlled gates [1].

3 Quantum Computing and Cryptography

The relationship between quantum computing and cryptography collides with one another because of the inherent nature of both fields. The purpose of cryptography is to securely send information over the internet and provide a key for the recipient to open the confidential information. The centerpiece of some widely used asymmetric cryptographic schemes are complex prime number problems that can only be solved using complex algorithms. These algorithms, when under the context of a classical computer, are not easy to break. But for quantum computing, its realization poses a major data security risk. The fundamental difference of a quantum machine that separates itself from classical computers is its ability to compute much more calculations than the latter, consequently, solving the traditional methods of key encryption with ease.

Due to this potential risk to data security because of quantum computers, post-quantum cryptography will address the safety concerns from these new innovations. There will be obstacles that the new form of cryptography will encounter, such as slower performances, larger computing resources, and non-standardized algorithms that will be used in the beginning of its inception. Although there are no quantum computers that could pose a security risk currently, the issue will inevitably come with the onset of quantum computing successfully opening to the public market.

However, quantum computing contributes not only to the termination of classical secure communication. The nature of quantum mechanics also delivers new innovations and improvements to cryptography.

3.1 Prime Factorization

When data is sent between two or more devices, the information is encrypted so that only parties that have the key can read its contents. In some widely used asymmetric cryptographic schemes such as RSA encryption, the key is knowing the prime factors of a very large number. Therefore, key generation consists of creating large enough numbers to combat decryption with the two prime number factors used as the key.

One of the leading applications of quantum computing in cryptography is the factoring the product of large prime numbers. With a quantum computer containing enough qubits, it is theoretically possible to factor the product of two large prime numbers like those used in encryption algorithms such as RSA, effectively bypassing their security. The only thing theoretically preventing this is the lack of a large enough quantum computer. The leading quantum prime factorization algorithm and a topic explored in this article is Shor's algorithm.

3.2 Quantum Key Distribution

Key distribution is a critical process in most cryptographic scheme. Interception of keys during the initial key exchange process can result in a compromised secure channel. The nature of quantum states provides a new process to generate random bit strings known between two party and prevent a third party from interception or influence of the key generation and distribution process.

3.3 Post-Quantum and XMSS

The term post-quantum refers to the technological period after quantum computers are advanced enough to break popular encryption schemes such as RSA, yet before quantum cryptography is developed. There will be a need for a post-quantum cryptographic signature scheme running on classical computers to maintain device security.

One proposed post-quantum scheme based on hash functions is the eXtended Merkle Signature Scheme (XMSS). Another proposed scheme is Leighton-Micali signature scheme (LMS) [8]. Any OpenSSL hash function can be used that uses Merkle-Darmgard construction for either of these schemes. XMSS has been shown to be highly efficient when hardware acceleration was used alongside software on a RISC-V SOC and on an FPGA for key generation [4][12].

4 Prime Factorization with Shor's Algorithm

Shor's algorithm is theoretically able to factor integers in polynomial time $\log(N)$, where N is the size of the integer. The best-known classical computing algorithm requires super polynomial time to factor the product of two primes, meaning the time needed can grow exponentially by the size of the integer. The idea of periodicity and the idea of factoring numbers are very much related and the simplicity to convert from one to another is why Shor's Algorithm is successful. Shor's algorithm uses modular exponentiation and the inverse Quantum Fourier Transform to find the period of the repetition of a signal which then using modular math can be used to find the factors of a number [1][3].

4.1 Relation of Prime Number to Asymmetric Key Cryptography

Many asymmetric key cryptography schemes rely on the complexity to compute certain problems without a certain piece of information, and often this information is prime number or were generated from prime number. For example, in

RSA encryption scheme, two keys, in the form of pair of positive integers, (e, n) and (d, n) are used and (e, n) is made public as public key. However, to generate the key, two prime number p and q are chosen such that $n = p \cdot q$, and d and e can then be generated by picking d to be a large random integer that satisfy the equation $gcd\,(d, (p-1)\cdot(q-1)) = 1$ and computing its modular multiplicative inverse e from d and $(p-1)(q-1)$ using a variation of Euclid's algorithm. Since d is also the modular multiplicative inverse of e, someone with the public key (e, n) can obtain the private key (d, n) by factoring n with Shor's algorithm to find p and q and solve for d using the same algorithm with e and $(p-1)(q-1)$ [10].

4.2 Mathematics in Shor's Algorithm

As stated earlier, Shor's Algorithm makes it possible to factor a semiprime number in polynomial time. It relies on the following fundamental equation:

$$g^x = m \cdot N + 1$$

where a given number g (the initial guess) raised to a specific power x equals a multiple of the semiprime number (the public key) plus 1. By modifying the equation, we get the following equation:

$$\left(g^{\frac{x}{2}} + 1\right)\left(g^{\frac{x}{2}} - 1\right) = m \cdot N$$

The value of $g^{\frac{x}{2}} \pm 1$ is potentially a factor of N or a multiple of N. In either case, the factors of N can be found immediately and the key decrypted. The problem with this algorithm is finding the value of x, the exponent of g. This can be run on a classical computer but takes an exponentially long time, whereas it can be run in polynomial time on a quantum machine. The quantum machine finds the period of the following function:

$$f(x) = a^x \pmod N$$

The sections below expand on quantum computations using Shor's Algorithm.

4.3 Quantum Fourier Transform

A quantum Fourier transform is effectively a change of basis from computational basis to Fourier basis followed by controlled rotations about the X-Y plane on the Bloch Sphere. This converts a binary number of 1's and 0's to their states with equivalent phases in the Fourier basis. This is a crucial function of quantum computing, because without this step the quantum computer would act only on 0's and 1's like a classical computer.

A quantum Fourier transform is accomplished in a quantum computer by using Hadamard gates and controlled phase shift gates on a set of qubits. First, a Hadamard gate is applied to the most significant qubit to change it from the computational basis to the Fourier basis. Several rotations are then applied to this qubit depending on the value of the lesser significant qubits; if the qubit is of value 1, the rotation is applied, if the qubit is of value 0, the rotation is not applied. The magnitude of this rotation depends on the index difference between the current qubit being rotated and the specific lesser significant qubit. When all lesser significant qubits are accounted for, the process is repeated on the next least significant qubit. The final output is reversed to represent the correct number as quantum computing syntax is reversed [1].

4.4 Inverse Quantum Fourier Transform

An inverse quantum Fourier transform is accomplished by reversing the steps of the quantum Fourier transform. The results of an inverse quantum Fourier transform are a sum of exponential terms. However, because of the properties of the results, they often destructively interfere with each other, producing only a single value. This destructive interference property of the inverse quantum Fourier transform makes it a vital part of many quantum algorithms a measure one result out of the infinite possibilities of quantum states [1].

4.5 Quantum Phase Estimation

Quantum phase estimation is the process of changing a quantum input of unknown phase from the Fourier basis to the computational basis so that the value can be measured. Measurement of this input cannot be taken directly because measuring quantum bits is done by taking the magnitude of the phases. This means that the angle of the phase will disappear. Quantum phase estimation first uses phase kickback to encode the phase of the input on control qubits using the unitary operator. Phase kickback is a phenomenon of quantum computing where the phase of a controlled qubit is imparted on the controlling qubit after an operation. An arbitrary odd number, a, that does not share a GCD with the number being factored, N, is exponentially modularized by repeated squaring, and each successive operator is controlled by the next most significant qubit. The value of the control qubits is then applied to an inverse quantum Fourier transform, where all incorrect values destructively interfere resulting in an answer of high probability of being correct. This result can be used easily in classical computers to factor the original input using modular math [1].

4.6 Classical Computation in Shor's Algorithm

The remaining portion of Shor's algorithm requires only simple modular math and can be done efficiently on a classical computer. The result obtained from the quantum phase estimation is divided by the number of control qubits used resulting in a number r. If the result r is even, perform the modular operation $a^{\frac{r}{2}} \pmod{N}$. If either the prior result r is odd, or the result of the modular operation is zero, repeat the entire algorithm. Next, find the GCD of the result of the modular operation ± 1. The results of this have a high probability of being factors of the original number [1][3].

4.7 Simulation of Shor's Algorithm

For the purpose of testing the validity of this algorithm to factor prime numbers, a simulation of this algorithm was run using qiskit and Jupyter Notebook. The simulation was done by first defining modules for modular multiplication, inverse quantum Fourier transform, and quantum phase estimation. Next an IBM token was used to connect to the IBM computer and the quantum simulator backend was called over the internet in order to simulate the quantum circuit. The simulation was done using 8 counting qubits and 4 register qubits. The algorithm was run until the factors were successfully found. After 503 trials of attempting to factor 15, 274 attempts were successful and 232 attempts were failures, yielding a success rate of 55%. It was found that the average amount of attempts needed for successful factorization was about 1.817 attempts. The time per attempt averaged around 1.41 seconds, timed by the python function perf_counter. This was done on a Windows 10 machine with an Intel i5-6600K @ 3.5GHz processor.

Further research may involve running the algorithm on an actual quantum computer instead of a simulation. In this case computation time may take longer overall due to the need to send the job over the internet and join the job queue. The amount of counting qubits can also be increased as well as the size of the number being factored in an attempt to further research success rate and computation time. However, currently the quantum computer with the maximum amount of qubits available to the public is 16, so expansion in this way isn't very possible.

5 Quantum Key Distribution

Key distribution is a critical process in symmetric key cryptography as it establishes the essential component, the shared key, shared between both parties that is then being used to encrypt and decrypt messages. One of the common problems with symmetric key cryptography is the possibility that during the initial establishment of this shared key, it may be influenced by a third party in a man-in-the-middle attack without being detected such that they can obtain the shared key to decrypt messages intended to be sent only to the original recipient or pose as either party by encrypting in the future.

As this attack requires receiving the secret key from the sender and sending a copy of the secret key to the original recipient, this attack can be avoided by establishing a secret key via quantum key distribution (QKD) processes using quantum states. Since an exact copy cannot be made per no-cloning theorem without prior knowledge of the states of the qubits and any attempt of doing so will result in the collapse of quantum superposition of the qubits, thus increase the quantum bit error rate (QBER) of the transmission and made apparent the existence of a third party [7].

5.1 BB84 QKD Protocol

The BB84 protocol was proposed by Bennett and Brassard to utilize polarization of photon along either rectilinear or diagonal basis for quantum key distribution purposes [2].

In such a process, the first user (Alice) attempt to establish a secret key shared with a second user (Bob). To prepare the photons, Alice first selects a random string of bits and a random string of choices between either rectilinear or diagonal basis of equivalent length to the string of random bits. Alice then encodes the photons into one of the four polarizations depending on the corresponding combination of bit and choice of basis and send it to Bob.

When Bob receives the photons, Bob will measure them with random basis for each photon independent of Alice's choice, which will result in a string of bits. Alice and Bob then reveal the bases they use to measure each proton. Upon learning about which bits are measured with the same basis for both parties, both parties confirm publicly with each other that a portion of bits obtained by measuring with matching basis matches each other's measurement with reasonable error; if not, then it is likely that the photons were tempered with. If it is determined securely transferred, then both parties retain the rest of the bits obtained by measuring with matching basis as shared secret.

5.2 Photon Number Splitting Attack

The integrity of the above key distribution scheme rests on the inability for a third party to obtain or produce an exact copy of photons sent. However, in a practical environment, the instruments sending out these photons may produce more than one photon with the same state at a time, allowing eavesdroppers to influence the process. For example, during the above QKD process between Alice and Bob, Eve may prevent any photons that does not have a copy when produced to be received by Bob and keep a duplicate of any photons it was produced with and send it to Bob. This way, Eve have an exact copy of what Bob received and therefore can obtain the shared key when the bases used for measurement is announced over public channel [7].

5.3 Decoy States

The use of decoy state addresses the vulnerability by sending decoy states with a different photon number than the signal states, which can be achieved by adjusting the intensity level of the instrument used to send out the photons. If Eve attempts the PNS attack, the difference in the fraction of photon number sent and detected will reveal her [7].

5.4 QKD Using Quantum Fourier Transform

Another method of key distribution, proposed by Huang and Chen, is to transform the classic bit string into frequency domain using QFT such that they are entangled [5]. "Decoy states" (different from the decoy states technique in the last section) in either X or Z basis will then be inserted randomly into the signal states before they are sent. PNS attack does not apply in this protocol as Eve cannot simply discard photons due to entanglement. Attempt to intercept, apply QFT to decode, and produce copy will not be successful as Eve does not know the position and basis of the decoy states to extract signal states to perform QFT. Alice and Bob can then verify the security and integrity by comparing measurement result of decoy states and digest of the classic bit string.

A concern of this protocol is that it may be prone to error in lossy and/or long channel. Different than the BB84 protocol, all states are needed to extract the original classic bit string. The author proposed the use of quantum error correction code technique to mitigate this problem, but transfer of long bit string may result in corruption of the correction code itself. Repeated transmission of the same classic string may be prone to PNS attack as it allows Eve to retrieve photons discarded previously. Splitting up long bit string and transmitting them in separate transmission may mitigate this issue but may reduce performance as it requires more rounds of verification. In comparison to BB84 with decoy states, this method provides an advantage of not having to adjusting intensity and counting photons but may not perform as well in terms of error and computing power needed due to use of QFT.

6 Conclusion

The purpose of post quantum computing and its applications to cryptography is essential to comprehend to prepare for the onset of quantum computing in the public market. The aspects of key distribution and prime factorization using Shor's Algorithm are crucial in understanding the applications of post quantum computing and how to apply it in real-life situations. A simulation of Shor's algorithm showed a high success rate of factoring the product of two prime numbers, 15 in this paper. The simulation resulted in a success rate of 55% with an average number of attempts needed being 1.817 at an average time per attempt of 1.41 seconds when run on an Intel i5-6600K @ 3.5 GHz processor. Further research using quantum computers and different parameters is needed to properly benchmark the algorithm. The onset of quantum computers poses a threat to the safety and security of data in communications because its ability to tackle complex security algorithms many compromise most classical computers' information, and post quantum cryptography is used to tackle the problems of future generations of computers.

References

1. Asfaw, A., Bello, L., Ben-Haim, Y., Bozzo-Rey, M., Bravyi, S., Bronn, N., Capelluto, L., Vazquez, A.C., Ceroni, J., Chen, R., Frisch, A., Gambetta, J., Garion, S., Gil, L., Gonzalez, S.D.L.P., Harkins, F., Imamichi, T., Kang, H., h. Karamlou, A., Loredo, R., McKay, D., Mezzacapo, A., Minev, Z., Movassagh, R., Nannicni, G.,

Nation, P., Phan, A., Pistoia, M., Rattew, A., Schaefer, J., Shabani, J., Smolin, J., Stenger, J., Temme, K., Tod, M., Wood, S., Wootton., J.: Learn quantum computation using qiskit (2020), http://community.qiskit.org/textbook

2. Bennett, C.H., Brassard, G.: Quantum cryptography: Public key distribution and coin tossing. Theoretical Computer Science 560, 7–11 (Dec 2014)
3. Blanda, S.: Shor's algorithm – breaking rsa encryption (Apr 2014), https://blogs.ams.org/mathgradblog/2014/04/30/shors-algorithm-breaking-rsa-encryption/
4. Buchmann, J., Dahmen, E., Hülsing, A.: XMSS - a practical forward secure signature scheme based on minimal security assumptions. In: Post-Quantum Cryptography, pp. 117–129. Springer Berlin Heidelberg (2011)
5. Huang, D., Chen, Z.: Quantum key distribution based on multi-qubit hadamard matrices. In: 2008 The Fourth International Conference on Information Assurance and Security. pp. 333–337. 2008 The Fourth International Conference on Information Assurance and Security (2008)
6. King, B.E.: Quantum state engineering and information processing with trapped ions. Ph.D. thesis, University of Colorado Boulder (Jan 1999)
7. Mailloux, L.O., Grimaila, M.R., Colombi, J.M., Hodson, D.D., Engle, R.D., McLaughlin, C.V., Baumgartner, G.: Quantum key distribution: examination of the decoy state protocol. IEEE Communications Magazine 53, 24–31 (Oct 2015)
8. McGrew, D., Curcio, M., Fluhrer, S.: Leighton-Micali Hash-Based Signatures. RFC 8554 (Apr 2019), https://rfc-editor.org/rfc/rfc8554.txt
9. Rieffel, E., Polak, W.: Quantum computing: a gentle introduction. Scientific and engineering computation, The MIT Press (2011)
10. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 21(2), 120–126 (feb 1978)
11. Voorhoede, D.: What is a qubit?, https://www.quantum-inspire.com/kbase/what-is-a-qubit/
12. Wang, W., Jungk, B., Wälde, J., Deng, S., Gupta, N., Szefer, J., Niederhagen, R.: XMSS and embedded systems. In: Lecture Notes in Computer Science, pp. 523–550. Springer International Publishing (2020)
13. Wilczek, F., Quanta Magazine moderates comments to facilitate an informed, s.: Entanglement made simple (May 2019), https://www.quantamagazine.org/entanglement-made-simple-20160428/