Quantum Computing: Computing of the Future Made Reality

Janelle Mathis
University of North Georgia, jnelly5500@gmail.com

Abstract

Abstract—Quantum computing is an emerging new area focused on technology consisting of quantum theory aspects such as electrons, sub-atomic particles, and other materials engineered using quantum mechanics. Through quantum mechanics, these computers can solve problems that classical computers deem too complex. Today the closest computing technology compared to quantum computers are supercomputers, but similarly to classical computers, supercomputers also have faults. With supercomputers, when a problem is deemed too complex, it is due to the classical machinery components within the computer, thus causing a halt in solving the task or problem. In contrast, these problems could be solved with a quantum computer due to the advancements in engineered materials based on quantum mechanics. Apart from the hardware that enables a quantum computer to function more intelligently, the software developed for these computers can also show tremendous improvements in certain aspects, such as cryptography. This research examines quantum computing from its origins and details how the computer runs, its faults and limitations, ways to protect from quantum computing attacks, and demonstrates what programming a quantum computer would entail.

Disciplines

Information Security | Management Information Systems | Technology and Innovation

Quantum Computing: Computing of the Future Made Reality

Janelle Mathis
Department of Cybersecurity
University of North Georgia-Dahlonega
jnelly5500@gmail.com
0009-0008-6570-6895

Bryson Payne
Department of Cybersecurity
University of North Georgia-Dahlonega
Bryson.Payne@ung.edu
0000-0003-4539-0308

***Abstract*—Quantum computing is an emerging new area focused on technology consisting of quantum theory aspects such as electrons, sub-atomic particles, and other materials engineered using quantum mechanics. Through quantum mechanics, these computers can solve problems that classical computers deem too complex. Today the closest computing technology compared to quantum computers are supercomputers, but similarly to classical computers, supercomputers also have faults. With supercomputers, when a problem is deemed too complex, it is due to the classical machinery components within the computer, thus causing a halt in solving the task or problem. In contrast, these problems could be solved with a quantum computer due to the advancements in engineered materials based on quantum mechanics. Apart from the hardware that enables a quantum computer to function more intelligently, the software developed for these computers can also show tremendous improvements in certain aspects, such as cryptography. This research examines quantum computing from its origins and details how the computer runs, its faults and limitations, ways to protect from quantum computing attacks, and demonstrates what programming a quantum computer would entail.**

***Index terms*—Quantum computing, qubit, lattice-based cryptography, QRev, QASM**

I. Introduction

Quantum computing was only known in the science fiction world for many decades until 1998. In 1998, Issac Chung, Neil Gershenfeld, and Mark Kubinec created the first quantum computer that could input data and output solutions but only for brief nanoseconds [1]. The same cannot be said for today's innovations in quantum computation; however, there is still more to be done. Although quantum computing is still a relatively new phenomenon, its applications for both the realm of science and the cybersecurity and computer science worlds can and will be unprecedented; however, with great power comes great responsibility. This research examines quantum computing from its origins and details how the computer runs, its faults and limitations, ways to protect it from quantum computing attacks, and demonstrates what programming a quantum computer would entail.

II. Background

Quantum computing is an emerging technology that makes use of quantum mechanics to solve problems that would otherwise be too complex for classical computers. The current computing technology that is most compared to quantum computing is that of supercomputers; however, supercomputers also have faults. When a supercomputer cannot solve a complex problem, it is typically due to the classical machinery within the supercomputer that is unable to either solve or process the problem, unlike quantum computing which can solve these complex problems with limited to no issue. Quantum computers use qubits to run the multidimensional quantum algorithms used to solve complex problems. Within a quantum computer are quantum processors, which must be very cold to operate. At IBM, quantum computers are kept at temperatures at a hundredth of a degree above absolute zero by using supercooled superfluids to, in turn, create superconductors. At this temperature, the electrons from the quantum computer can move through the processor without any resistance to form copper pairs. These copper pairs can then carry the charge across barriers or insulators through quantum tunneling. On another note, when two superconductors are placed on either side of the insulator, the Josephson junction is formed. IBM quantum computers use these Josephson junctions as superconducting qubits. More specifically, the firing of microwave photons at these qubits will allow for the control of the qubit's behavior and allow them to hold, change, and read out individual pieces of quantum information.

In placing the quantum information on a qubit, the qubit is held into a superposition state which "represents a combination of all possible configurations of the qubit" [2]. When a group of qubits is in this superposition state they can create complex, multidimensional computational spaces that a classical computer could never create. Finally, for a quantum computer, the topic of entanglement also can occur. Entanglement is the quantum mechanical effect that shows the correlation between the behavior of two different things. More specifically, if two qubits were entangled, changes to one of the qubits directly impact the other. The algorithms within the quantum computer monitor and manage these relationships to find the solutions needed to solve complex problems.

III. Quantum vs RSA

Even though quantum cryptography sounds like a threat or may seem more protective, RSA cryptography is most commonly used today. RSA cryptography is an asymmetric cryptography scheme that utilizes keys for encryption. More specifically, for encryption, a public key is required, while for decryption, one must possess a private key. The RSA cryptographic algorithm utilizes what is known as a totient function to create encryption and decryption [3]. A totient function is defined as the number of positive integers that are greater than or equal to the variable n that are relatively prime, in which both integers do not share any factors in common, to n, where one is counted as being relatively prime to all numbers [4]. To try to break this encryption using modular exponentials requires heavy computation, as the most direct way to crack this encryption is to find all prime factors of the variable n. However, Shor's algorithm utilizes quantum mechanics to make this computation significantly easier and takes some time off. This is because the algorithm is a quantum algorithm for finding prime factors of an integer or, in this case, the variable n.

Even with RSA cryptography, there is always the need for innovation which is why Post-Quantum Cryptography is being brought into the conversation more and more today. Post-quantum cryptography is a research field that develops classical asymmetric encryption schemes that are not vulnerable to classical or quantum attacks. Along with this, quantum key distribution is also coming to fruition. The quantum key distribution uses quantum physics to exchange the encryption keys whose security is intertwined with the laws of physics rather than with computational power, making the keys unconstructive if an attack occurs.

IV. Protections

In addition to quantum key distribution, other systems and tools have been initiated to help protect quantum computers and information, such as lattice-based cryptography. Lattice-based cryptography is based on the idea that a cryptographic scheme is built on mathematical problems around lattices [5]. For lattice-based cryptography, the lattice would resemble a grid on graph paper using a set of points, which are vectors, located at the intersection of straight lines that run infinitely [5]. These vectors are then added and subtracted by any integer multiples [5]. An issue that can occur is finding points within a lattice close enough to zero or close to another point. These points are the private and public keys used for encryption. One set of points would act as the private key, and another would serve as the public key; however, the public key's location would be farther apart from the private key's set of points. For these points to be found, a brute force search including all the possibilities would need to occur to distinguish the private key from the public key, which would take a considerably large amount of time for classical computers [5]. For a quantum computer, it is thought to be unable to solve these lattice-based problems in a reasonable period; however, it would still be faster than a classical computer [5]. Three major lattice-based algorithms and encryption systems are used today to demonstrate lattice-based cryptography: CRYSTALS-Kyber, CRYSTALS-Dilithium, and NTRU.

CRYSTALS-Kyber is an IND-CCA2 secure key encapsulation mechanism (KEM) based on the difficulty of solving the learning with errors (LWE) problem, the solving of linear equations with noise, over the mathematical concept of module lattices [6],[7]. IND-CCA2 stands for indistinguishability of ciphertexts under adaptive chosen attacks [8]. In an IND-CCA2 attack, a user tries to decipher ciphertext by making queries to a decryption oracle which runs a decryption algorithm with a key possessed by an attacker from earlier in the decryption process [9]. In this attack, the attacker knows the ciphertext when making the queries to the decryption oracle, whereas in an IND-CCA1 attack does not know the ciphertext before making the queries to the decryption oracle [9]. An example of this would be "if a server that can decrypt and parse through a deciphered plaintext program then produces a message and digital signature as a pair and checks this pair against a public key that is unrelated to the encryption key with no issues the server will act out the message from the message and signature pair" [9]. For an algorithm such as CRYSTALS-Kyber to be listed as secure in regards to IND-CCA2, the system, according to NIST, "is secure if no adversary can distinguish "challenge encryptions" of two messages of their choosing, despite having oracle access to both encryption and

decryption" [8]. As stated earlier, Kyber is rooted in the learning-with-error sapproach, or LWE-based encryption, which has been improved by applying ideas from another lattice-based system NTRU to define the different phenomenons, Ring-LWE and Module-LWE [6]. With this being said, Kyber is built on top of a chosen-plaintext-attack secure cryptosystem based on the difficulty level of Module-LWE [6].

CRYSTALS-Dilithium is a digital signature lattice-based algorithm that is firmly secure under chosen message attacks based on the level of difficulty of lattice problems over module lattices [9]. This scheme means that an attacker with access to an oracle that can sign cannot produce a digital signature of a message whose digital signature has yet to be seen by the system [10]. This also applies to an oracle that tries to produce a different signature for an already existing message that has already been signed. The design for Dilithium is based on a technique used by Vadim Lyubashevsky, the Fiat-Shamir with Aborts technique which uses rejection sampling to secure the lattice-based schemes [10]. The Fiat-Shamir with Aborts technique combines the Fiat-Shamir heuristic and the concept of aborting. The Fiat-Shamir heuristic transforms a proof system based on interactive public coins into a digital signature by replacing the public coins with hash function evaluations [11]. Vadim Lyubashevsky proposed a lattice-based signature scheme that resembles another instance of the Fiat-Shamir heuristic, Schnorr's signature, which relies on the discrete logarithm problem [11]. The difference between Schnorr's signature and Lyubashevsky's signature is that in Lyubashevsky's signature, "the underlying interactive proof system has a non-negligible probability of aborting" [11]. Aborting is the concept that allows signature distribution to be separate from its signing key [11]. Aborting is also necessary to avoid attacks targeting signature schemes such as Schnorr's signature. To deal with the aborts, the protocol is executed repeatedly within a loop until no aborts occur in the most recent loop [11]. CRYSTALS-Dilithium improves on the current most efficient scheme that only uses the uniform distribution rather than the Gaussian, or normal, distribution by utilizing a new technique that reduces the public key by more than a factor of two [10].

NTRU is a public key cryptosystem that utilizes lattice theory for the encryption and decryption of data. NTRU is based on the algebraic structures of specific polynomial rings that are embedded with messages [12]. This system consists of NTRUEncryt, designed for encryption, and NTRUSign, designed for digital signature, with the former being the most utilized for quantum security. NTRU relies on the closest vector problem and the shortest vector problem rather than using large prime numbers like RSA cryptography. The closest vector problem is a computational problem that finds vectors relative to a target vector or vectors in a space, while the shortest vector problem's objective is to find the shortest non-zero vector within a given lattice [13]. The closest vector problem essentially is a generalization of the shortest vector problem. However, the closest vector problem is more challenging than the shortest vector problem. Regardless of this, both lattice problems are crucial for lattice-based cryptography. With NTRU encryption, public key encryption enables high-speed processing and performs encryption and decryption using polynomial operations implemented at a higher speed [12]. Compared to Kyber, NTRU is slower regarding encryption and decryption, and unlike Kyber and Dilithium, NTRU is also not IND-CCA secure in its public key encryption [8]. With this, there is still at least one impactful benefit of using NTRU: NTRU is still faster than RSA cryptography. Still, most importantly, NTRU is resistant to quantum computing attacks.

V. Reverse Engineering

With quantum computing now being put into conversations with our current systems, many aspects can and should be explored. More specifically, the topic of software reengineering can and will be vital for the future of computation. Even within software reengineering, reverse engineering will be utilized. With this being a relatively new concept and programming strategy, more information must be disclosed, and more studies must be conducted. In 2021, a study was done to demonstrate the beginning of what reverse engineering could look like on a quantum computer. Ricardo Pèrez-Castillo, Luis Jimènez-Navajas, and Mario Piattini began work on a reverse engineering tool named QRev. QRev generates intricate/abstract representations of quantum programs that can be introduced to and work with other elements that originated from classical information systems [14]. For the study, QRev was used to generate Knowledge Discovery Models (KDM) from a quantum source code written in Q# [14]. It is important to note that a Knowledge Discover Model is "a metamodel that defines the common vocabulary of knowledge related to software engineering artifacts, regardless of the implementation programming languages and runtime platform, and is designed to enable knowledge-based integration between tools" [15]. In other words, in this study, KDM is used to demonstrate the effectiveness of using QRev for converting or rewriting programs from an older language/system to the current programming language/system.

QRev can be broken into two modules about the Q# programming language to better understand this process. The first module is the Q# parser, which analyzes Q# files while considering grammar, such as words and phrases, allowing Q# to acknowledge and recognize Q# elements to build the abstract syntax tree used for the second module later. It is important to note that for QRev to be implemented, it first had to be developed as a REST API to show the service operations employed to transform the Q# files and generate the KDM model files. The Q# parser was developed using ANTLR, a powerful parser generator used for reading, processing, executing, and translating structured text or binary files. The grammar of the Q# syntax is broken into two elements, the lexer, and the actual parser. The lexer defines the lexicon, a book containing the alphabetical arrangement of words in a language and their definitions and reserved words of the language. The parser specifies the structure of the language. For the study, an ANTLR project for C# was adapted into Q# by defining the necessary keywords, structures, and modules since Q# is based on C#. The second module is the KDM generator which takes the abstract syntax tree generated from the Q# parser to generate the model. The KDM generator receives the abstract syntax tree generated by the Q# parser as input and then inspects the tree node by node using the JDOM library, an open-source library implemented by Java to analyze XML documents. The generator manages all the elements from the syntax tree and their references in hash maps to define relationships amongst KDM elements.

For the study, two questions were posed and answered. The first question is if QRev can generate accurate and complete KDM models. In the study, five programs were parsed, which means the Q# parser was victorious at a 100% rating. Additionally, five KDM models were generated as well. The combination of precision and recall into one score was computed to test the true success of QRev, precision, recall, and F-measure. For these variables to be computed, numerous elements were counted, such as the total number of lines of source code for each program, the number of quantum elements in source code (counted as the expected relevant elements), the number of KDM elements, the number of missing elements in the KDM models (resulting in false negatives), and the number of irrelevant elements in the KDM models (resulting in false positives) [14]. According to the study, both the values for precision and recall were high, which shows that the KDM generator is also effective. The F-measure was 0.92, which means the effectiveness of QRev is 92%.

The second is if QRev can generate the KDM models in a scalable manner. According to the study, all five programs could be transformed into KDM models successfully. Additionally, the models were generated in less than one second, so QRev could be applied to real environments and be efficient at a reasonable time. It should also be essential to note that most of the time of the study was spent parsing the Q# files and building the abstract syntax tree, while the KDM generation from the abstract syntax tree took only 4% of the total time.

VI. Implementation

In the research examined, the assembly code known as QASM is shown to have a direct correlation to quantum circuits which help visualize the algorithms used within a quantum computer. With this direct correspondence, reading a script from top to bottom in a code script also allows the code script to be read left to right within a circuit. This is the best visualization of how circuits in a quantum computer operate.

A. Source Code Breakdown

In Fig. 1. a QASM executable file was created to show a three-bit random number generator that would be able to be seen in a quantum computer's circuit as well as its circular notation. This would demonstrate that for every qubit that is listed a number between the range of "000" to "111" would be generated and stored within its corresponding classical register.

1) QASM Version and Include Command: The OPENQASM 2.0 command is calling for the specific QASM version with the include command calling for extra gates to be included in the program.

2) Creation of Qubits and Classical Bits: The qreg (short for quantum register) q[3] and creg (short for classical register) c[3] commands create three qubits and three classical bits for the program.

3) Quantum Instructions: The commands h q[x] contain a gate, H, and a corresponding qubit register, q[0] to q[2]. Gates change the state of qubits, and when several of these instructions are applied together, an algorithm is formed with quantum circuits helping to visualize these newly formed algorithms. The H gate, or the Hadamard gate, is a quantum gate that creates superposition states for a qubit. The quantum instructions that are being used are calling for the three H gates to be applied to the first three qubits.

4) Measurement: The measure command decides which qubit is to be measured and which classical register to store that bit value in. In quantum computing, measurement is the process of reading information encoded within quantum systems. For the specific command in the source code, all the defined

qubits are to be measured and stored in the corresponding classical registers.

```
1   OPENQASM 2.0;
2   include "qelib1.inc";
3
4   qreg q[3];
5   creg c[3];
6
7   h q[0];
8   h q[1];
9   h q[2];
10
11  measure q->c;
```

Fig. 1. Source code of QASM to create a three-bit random number generator listed in increasing qubit order and to store the results from the qubits to the corresponding classical registers.

VII. Conclusion

Although it is still relatively new, quantum computing will offer many new opportunities to better advance computer science and cybersecurity. This research examined only a few aspects quantum computing has strived to accomplish. From understanding the creation of quantum computers to the different systems developed to protect against quantum attacks, the opportunities for quantum computing are only just arising. More opportunities can appear in the future in areas such as random number generation, communication, cryptography, machine learning, and many more. It is safe to say quantum computing has been made a reality.

References

[1] W. C. Holton, "Quantum computer," Encyclopædia Britannica, https://www.britannica.com/technology/quantum-computer (accessed Dec. 3, 2021).

[2] "What is quantum computing?," IBM, https://www.ibm.com/topics/quantum-computing (accessed Apr. 28, 2023).

[3] N. Berner, "The attack on RSA with a quantum computer," Breaking Security Protocols with a Quantum Computer, https://www.scip.ch/en/?labs.20220519 (accessed Apr. 28, 2023).

[4] E. W. Weisstein, "Totient function," from Wolfram MathWorld, https://mathworld.wolfram.com/TotientFunction.html (accessed Apr. 28, 2023).

[5] "What is lattice-based cryptography?," Utimaco, https://utimaco.com/products/technologies/post-quantum-cryptography/what-lattice-based-cryptography (accessed Jul. 19, 2023).

[6] P. Schwabe, "Crystals," Kyber, https://pq-crystals.org/kyber/index.shtml (accessed Jul. 19, 2023).

[7] D. Balbás, "The hardness of LWE and Ring-LWE: A survey - IACR," Cryptology ePrint Archive,

[8] G. Alagic *et al*., "Status report on the third round of the NIST post-quantum cryptography standardization process" National Institute of Standards and Technology, https://nvlpubs.nist.gov/nistpubs/ir/2022/NIST.IR.8413-upd1.pdf (accessed Jul. 19, 2023).

[9] F. Onstrike, "The essential differences between ind-CCA1 and ind-CCA2?," Cryptography Stack Exchange, https://crypto.stackexchange.com/questions/99295/the-essential-differences-between-ind-cca1-and-ind-cca2 (accessed Jul. 19, 2023).

[10] P. Schwabe, "Crystals," Dilithium, https://pq-crystals.org/dilithium/index.shtml (accessed Jul. 24, 2023).

[11] J. Devevey, P. Fallahpour, A. Passelègue, and D. Stehlé, "A detailed analysis of Fiat-Shamir with aborts - IACR," Cryptology ePrint Archive, https://eprint.iacr.org/2023/245.pdf (accessed Jul. 24, 2023).

[12] A. P. Premnath, "Application of NTRU cryptographic algorithm for securing SCADA communication," UNLV University Libraries,

[13] T. Wunderer, "On the security of lattice-based cryptography against lattice reduction and Hybrid Attacks," Core, https://core.ac.uk/download/pdf/162020363.pdf (accessed Dec. 3, 2021).

[14] R. Pérez-Castillo, L. Jiménez-Navajas, and M. Piattini, "QRev: migrating quantum code towards hybrid information systems - Software Quality Journal," SpringerLink, https://link.springer.com/article/10.1007/s11219-021-09574-x (accessed Apr. 28, 2023).

[15] "Overview of the OMG knowledge discovery metamodel (KDM) spec," KDM Analytics, https://kdmanalytics.com/resources/standards/kdm/ (accessed Apr. 28, 2023).