Quantum Advantage and Y2K Bug: Comparison

Lei Zhang, Andriy Miranskyy, and Walid Rjaibi

Abstract—Quantum Computers (QCs), once they mature, will be able to solve some problems faster than Classic Computers. This phenomenon is called "quantum advantage" (which is often used interchangeably with a stronger term "quantum supremacy"). Quantum advantage will help us to speed up computations in many areas, from artificial intelligence to medicine. However, QC power can also be leveraged to break modern cryptographic algorithms, which pervade modern software: use cases range from encryption of Internet traffic, to encryption of disks, to signing blockchain ledgers.

While the exact date when QCs will evolve to reach quantum advantage is unknown (the forecasts range between months and a decade), the consensus is that this future is near. Thus, in order to maintain crypto agility of the software, one needs to start preparing for the era of quantum advantage proactively (before the software and associated data are compromised).

In this paper, we recap the effect of quantum advantage on the existing and new software systems, as well as the data that we currently store. We also highlight similarities and differences between the security challenges brought by QCs and the challenges that software engineers faced twenty years ago while fixing widespread Y2K bug. Technically, the Y2K bug and the quantum advantage problems are different: the former was caused by timing-related problems, while the latter is caused by a cryptographic algorithm being non-quantum-resistant. However, conceptually, the problems are similar: we know what the root cause is, the fix (strategically) is straightforward, yet the implementation of the fix is challenging.

To address the quantum advantage challenge, we create a seven-step roadmap, deemed 7E. It is inspired by the lessons-learnt from the Y2K era amalgamated with modern knowledge. The roadmap gives developers a structured way to start preparing for the quantum advantage era, helping them to start planning for the creation of new as well as the evolution of the existent software.

1 Introduction

The field of quantum computing is still young, but it has been evolving rapidly during the last decade. The power of quantum computing will threaten modern cybersecurity platforms. Thus, we position that the software engineering community should start thinking about the impact of quantum computing on cybersecurity and the best practices to address these concerns. Let us look at the evolution of Quantum Computers (QC).

In 1982, Feynman introduced the idea of quantum computing [1]; Shor proposed the first practically relevant algorithm (for breaking encryption protocols based on integer factorization and discrete logarithm) that can be efficiently computed on a QC in 1994 [2].

It took a while to implement an actual QC. A partnership between academia and IBM created the first working 2-qubit QC in 1998 [3], but it took the company 18 years to make a 5-qubit QC accessible to the public in 2016 [4].

At present, a few QCs are commercially available. DWave started selling adiabatic QC in 2011 (although the debate about adiabatic QC being a "true" QC is ongoing[1] [6]) with the current offerings having $>$ 2000 qubits. IBM gave access to 20- and 50-qubit gate-based superconducting QCs to academic and industrial partners to explore practical applications in 2017 [7].

For non-commercial use, IBM offers 5- and 14-qubit QCs via IBM Q Experience online platform based on IBM Cloud (along with local- and Cloud-based simulators) [8]. Microsoft provides access to a simulator of a topological QC via Microsoft Quantum Development Kit [9] (and is planning to give access to an actual QC in the future). Google built 72-qubit gate-based superconducting QC in 2018 [10], but it is not publicly accessible at the time of writing.

It is said that in the future, a QC can solve some problems much faster than a Classical Computer (CC), which is called quantum advantage [1] (often used interchangeably with the term quantum supremacy, which denotes an ability of QCs to solve problems that CCs cannot). This is because QC compute power is growing faster than CC one. The growth of CC power was long governed by Moore's Law [11], i.e. the power of CCs doubling every two years (an exponential increase — $O(2^m)$, where m is the number of years). The pace has slowed recently to doubling every 2.5 years [12], yet remains strong. Hartmut Neven, the founder and manager of the Quantum Artificial Intelligence Lab at Google, claimed that quantum computing power follows more dramatic growth rate, of double exponential growth $O\left(2^{2^m}\right)$. If this is the case, the quantum advantage can be achieved in a matter of months [13].

This is not a definitive prediction: the exact growth rate of QCs is under debate. Norishige Morimoto, the director of IBM research in Tokyo and global vice president at IBM, claims that quantum advantage will be achieved between years 2022–2024 [14]. Michelle Mosca, a co-founder of the Institute for Quantum Computing and chief executive of evolutionQ, claims that QC may outperform CC in certain tasks after 2026 [15].

Based on the above, we can conclude that while the exact date of achieving quantum advantage is not known, the general consensus is that the QCs are quickly developing and may become practical within the next decade.

- *L. Zhang and A. Miranskyy are with the Department of Computer Science, Ryerson University, Toronto, Canada.*
  E-mails: leizhang@ryerson.ca and avm@ryerson.ca
- *W. Rjaibi is with IBM Canada Lab, Toronto, Canada.*
  E-mail: wrjaibi@ca.ibm.com

[1] A hybrid of adiabatic and gate-based QC is promising [5], but no commercial implementation is available.

With the vast increase in computing power, QCs promise to revolutionize many fields, including artificial intelligence, medicine, and space exploration [16]. But they can also be abused to break key encryption algorithms the Internet depends upon today for ensuring the safety and privacy of digital information. This can be achieved by speeding up 1) factorization of integers, solving the discrete logarithm problem, and the elliptic-curve discrete logarithm problem (using Shor's algorithm [2]); as well as 2) the search in a set (with the help of Grover's algorithm [17]). Both tasks are foundational for modern encryption algorithms. Let us elaborate.

2 The impact of quantum computing on cybersecurity

It is essential to understand that quantum computing will affect encryption differently depending on the class of encryption algorithm.

Below we first elaborate on the asymmetric encryption algorithms, which are used in many areas ranging from the Transport Layer Security (TLS) protocol (used to safeguard data passed between two systems on the Internet) to Pretty Good Privacy (PGP) software used to encrypt and decrypt a file and safely transfer it between computers.

We then proceed to the symmetric encryption algorithms, such as Advanced Encryption Standard (AES), used to protect sensitive data. There exist numerous use-cases, ranging from encrypting a file archive (e.g., implemented in 7z and WinZip software) to encrypting computer's disks (e.g, using Apple MacOS FileVault and Symantec Endpoint Encryption).

2.1 Asymmetric encryption

Asymmetric encryption algorithms, which are based on factoring large integers (e.g., Rivest–Shamir–Adleman — RSA), discrete logarithms (e.g., Elliptic Curve Cryptography — ECC, and Diffie-Hellman key exchange), or similar approaches (see [18], [19] for review) will need to be replaced by quantum-resistant alternatives [18], [19]. Effective security strength, shown in Table 1, suggests that the strength of the RSA and ECC is somewhat weaker or comparable to AES on a CC, but is extremely weak on a QC. This is because Shor's algorithm [2] can perform integer factorization in polynomial time; so what requires thousands of years with classical computers would only take days/hours on a large-scale quantum computer. This, of course, assumes that a large-scale quantum computer with the required number of qubits exists, which is not the case right now.

2.2 Symmetric encryption

Unlike asymmetric encryption algorithms, symmetric encryption algorithms do not face an existential threat: one needs to perform a brute-force attack to break it. However, on a classic computer generation of n keys require $O(n)$ operations, while on a QC it can be done using $O\left(\sqrt{n}\right)$ operations, thanks to Grover's algorithm. Thus, a large quantum computer running Grover's algorithm could provide a quadratic improvement in brute-force attacks on symmetric encryption algorithms, such as AES. This translates into a need to double key size to support the same level of protection. For AES specifically, this means using 256-bit keys to maintain today's 128-bit security strength[2], as depicted in Table 1.

3 The impact on security-critical systems

3.1 Newly-built systems

3.1.1 Asymmetric encryption

For new systems involving a security component, practitioners will have to replace modern asymmetric algorithms (e.g., RSA) with those that are based on algebraic operations which QC cannot perform efficiently (in comparison with a CC). The field of post-quantum (also known as quantum-resistant) cryptography deals with such algorithms. Examples of the principles include but are not limited to lattice-based cryptography (e.g., used in NTRU [20] and BLISS [21] cryptographic schemes) and hash-based cryptography (e.g., using Merkle Hash Tree signature [22]).

The efforts are underway to introduce cryptographic standards for the era of quantum advantage. Currently, NIST is running a competition to select the best quantum-resistant algorithm. In January of 2019, they announced that the first round of the NIST Post-Quantum Cryptography Standardization Process was completed by selecting 26 contestant algorithms, which will be further tested in the second round of the process [23].

Thus, soon, a practitioner may be able to leverage a standardized algorithm right away, while designing new software. For now, when creating a new software product that is expected to have a lifespan long enough to be affected by the quantum advantage, we advise designing security component in such a way that the underlying cryptographic algorithm can be replaced with a different one.

3.1.2 Symmetric encryption

If the system requires a symmetric algorithm, then one can leverage a standard implementation of an existing algorithm (e.g., AES). In this case, the component has to be designed in such a way that it can accommodate the increase of the key length for a given algorithm.

3.2 Threats to the existing data

While large-scale quantum computers might be several years away, someone with malicious intent could still capture sensitive encrypted data (e.g., by capturing encrypted network packets protected with TLS or by cloning an encrypted disk), then store that data somewhere in a data lake. When a large-scale quantum computer becomes available, this person can leverage QC power to break the asymmetric encryption used by TLS or brute-force access to the encrypted disk and recover the sensitive data. While not much can be done about the protocols involving asymmetric algorithms; for the symmetric ones, we can increase the length of the key right away to cumber the brute-force attack [24]. We can also encrypt archived data (e.g., stored on backup devices) with a quantum-resistant algorithm.

[2] In other words, an n-bit AES cipher provides a security level of $n/2$ because $\sqrt{2^n} = 2^{n/2}$.

TABLE 1
Effective security strength of key encryption algorithms as per [18]

Encryption algorithm	Key size (bits)	Effective security level on CCs (bits)	Effective security level on QCs (bits)
RSA 1024	1024	80	0
RSA 2048	2048	112	0
ECC 256	256	128	0
ECC 384	384	256	0
AES 128	128	128	64
AES 256	256	256	128

Another example is a blockchain platform using proof-of-work algorithms. The security of current blockchain platforms relies on a digital signature, which is based on either Elliptic Curve Digital Signature Algorithm [25] or RSA algorithms; both are vulnerable to QCs. Kiktenko et al. [26] proposed a quantum-secured blockchain framework that utilizes Quantum Key Distribution techniques via an experimental fibre network (the cost of the network is not disclosed). An alternative is to introduce a quantum resistant asymmetric algorithm and recompute the Nonce for all legacy blocks using new algorithms, which may be expensive. A more efficient approach may be to switch to a proof-of-stake algorithm from a proof-of-work algorithm.

3.3 Legacy systems

If the legacy system is well-designed and actively maintained, then the solution is straightforward: one can replace an existing asymmetric algorithm with a new one (or increase a key size of an asymmetric one) while ensuring that the existing data can be migrated to a new format. However, it may require a downtime to re-encrypt existing data, and re-encrypting data is typically disruptive until the new encrypted data are available again.

The software may run on antiquated hardware that does not have sufficient computing power to run QC-resistant algorithms. In this case, we may need to upgrade this obsolete equipment or virtualize the outdated runtime environment, so that it can be executed by a hypervisor on a more powerful computer.

Often, altering existing (legacy) system to address the security concern may be challenging. Legacy systems frequently lack adequate information or support to be maintained or upgraded [27]. The root causes of these issues are numerous. For example, developers of the system may be unavailable (e.g., because they left the company or retired), source code or documentation may be lost, or build platforms for the source code may be sunset. To make matters worse, the encryption-related code may be spread or cloned among multiple software components (due to bad design), making alterations even more challenging. These root causes make it extremely difficult and expensive to upgrade such a system to the newest security protocols, making the replacement the only feasible option.

4 Quantum advantage and Y2K bug: parallels

All of these challenges, conceptually, pose a striking similarity to the Y2K bug [28], [29], which happened around the year 2000. The root cause of the Y2K bug was because older software represented four-digit years with only the last two digits, while the first two digits were fixed at 19. That is an increment by 1 of the year 1999 would result in the year 1900 rather than the year 2000.

Obviously, the underlying root cause of the Y2K and the quantum advantage problems are different. However, conceptually, the solutions to the problems are similar. In the Y2K case, we had to take the timing-related code and replace it with a new one capable of representing years using 4 digits. In the quantum advantage case, we will have to take an encryption-related code and either replace the algorithm (asymmetric case) or increase key length (symmetric case).

Another difference lies in the fact that failures associated with the Y2K-related defect were encountered after a particular date, namely, December 31, 1999. In the case of quantum advantage, the exact date is unknown. However, we may still leverage lessons learnt from fixing legacy systems to fix the Y2K bug. Let us review what we have learned from Y2K.

Putnam and Myers [30] divided the legacy systems into three groups for the Y2K bug. The same grouping strategy can be adopted for the quantum advantage problem: 1) work first on those that involve life and death, 2) work on those that are critical to the continued operation of your organization, and 3) work on those in which security is merely irritating, not costly.

As the era of quantum computing is approaching, some actions can be taken now. Shimeall et al. [31] proposed several guidelines for the security concerns before Y2K, some of which are still applicable to our current situation: 1) existing systems must be examined and repaired, and 2) programmers and designers must be educated about the new security challenges brought by quantum computers. The second guideline applies to the creation of the new systems too.

Schultz [32] described five steps that a software engineer should follow to respond to the Y2K bug, which we can utilize in our challenge: 1) gain senior management's acknowledgement of the potential impacts, 2) assess the problem and alternative solutions, 3) estimate the cost and gain approval of selected solutions, 4) execute and control the solutions as a partner with senior management, and 5) monitor solutions' early results in production.

5 Roadmap

We propose a **7E** roadmap for software developers, summarizing steps needed to address encryption-related challenges associated with quantum advantage. Our 7E roadmap consists of the following seven steps:

1) **E**ngage executives and senior management so that they can sponsor the initiative. It is important to get the acknowledgement from the decision makers in your company or organization. Moreover, executives and senior management can assess security concerns from a different perspective. To put them in context, you can use formal presentations or reports and incorporate feedback from them later on.
2) **E**xamine existing products and their cybersecurity components to identify and locate the issues, review the document and/or programs and assess the problems. For legacy systems, there exist difficult scenarios, such as lack of documentation, source code, or build infrastructure. Identify existing data (if any) that may requires protection.
3) **E**volve: design a new software with crypto agility in mind, so that quantum-resistant algorithm can be added to the software later on. For example, the encryption component can be designed to be plug-and-play, so that an existing encryption algorithm can be replaced (if it is discovered to be vulnerable) with a robust one. This will save costs in the future, when the standards for quantum-resistant encryption algorithms are finalized and our software has to be updated with these algorithms.
4) **E**ducate the programmers and designers to make sure that everyone is 'on the same page' because (in most cases) the security-related component are coupled with the remaining software components. This implies that the whole development organization needs to be aware of the challenges associated with quantum advantage.
5) **E**stimate the impact of potential problems and the cost of alternatives to prioritize the problems. Rate the cost of potential solutions in terms of human and time resources. Work first on the systems that handle critical data first. The definition of critical will vary from industry to industry. A representative example is a system handling personal data, such as financial transactions or health records.
6) **E**xecute the new cybersecurity policy. Select and adopt appropriate solutions based on requirements, budgets, and priorities. As discussed in Section 3, practitioners can execute the new cybersecurity policy in different ways. For newly-built systems, post-quantum cryptography may be adopted (see Section 3.1). For legacy systems, the software and associated hardware may have to be altered (see Section 3.3). For existing data, an intermediate solution — e.g., re-encryption — may be applied (see Section 3.2).
7) **E**ssay the new cybersecurity policy. Keep monitoring the performance and the robustness of your new cybersecurity policy in production to make sure that the challenges associated with quantum advantage were addressed; adjust the policy if needed.

6 Summary

To summarize, the community of software practitioners needs to start preparing for the era of quantum advantage on three fronts: 1) developing new software with new encryption algorithms in mind, 2) upgrading the legacy software with new encryption algorithms, or 3) migrating the business processes to more modern software if legacy software cannot be upgraded. The earlier we start — the higher the chances that we will be able to address proactively security challenges that this era brings with it.

References

[1] R. P. Feynman, "Simulating physics with computers," *International journal of theoretical physics*, vol. 21, no. 6, pp. 467–488, 1982.

[2] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM J. Comput.*, vol. 26, no. 5, pp. 1484–1509, Oct. 1997. [Online]. Available: http://dx.doi.org/10.1137/S0097539795293172

[3] I. L. Chuang, N. Gershenfeld, and M. Kubinec, "Experimental implementation of fast quantum searching," *Phys. Rev. Lett.*, vol. 80, pp. 3408–3411, Apr 1998. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.80.3408

[4] "IBM News room - 2016-05-04 IBM Makes Quantum Computing Available on IBM Cloud to Accelerate Innovation - United States," accessed on 2019-07-19. [Online]. Available: https://www-03.ibm.com/press/us/en/pressrelease/49661.wss

[5] R. Barends, A. Shabani, L. Lamata, J. Kelly, A. Mezzacapo, U. L. Heras, R. Babbush, A. G. Fowler, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, E. Jeffrey, E. Lucero, A. Megrant, J. Y. Mutus, M. Neeley, C. Neill, P. J. J. O'Malley, C. Quintana, P. Roushan, D. Sank, A. Vainsencher, J. Wenner, T. C. White, E. Solano, H. Neven, and J. M. Martinis, "Digitized adiabatic quantum computing with a superconducting circuit," *Nature*, vol. 534, pp. 222–226, Jun. 2016. [Online]. Available: http://dx.doi.org/10.1038/nature17658

[6] T. Albash, V. Martin-Mayor, and I. Hen, "Temperature scaling law for quantum annealing optimizers," *Phys. Rev. Lett.*, vol. 119, pp. 110 502:1–110 502:7, Sep 2017. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.119.110502

[7] "IBM Announces Collaboration with Leading Fortune 500 Companies, Academic Institutions and National Research Labs to Accelerate Quantum Computing - Dec 13, 2017," accessed on 2019-07-19. [Online]. Available: https://newsroom.ibm.com/2017-12-13-IBM-Announces-Collaboration-with-Leading-Fortune-500-Companies-Academic-Institutions-and-National-Research-Labs-to-Accelerate-Quantum-Computing

[8] "IBM Q Experience," accessed on 2019-07-19. [Online]. Available: https://quantumexperience.ng.bluemix.net/qx/experience

[9] "Quantum computing | Microsoft," accessed on 2019-07-19. [Online]. Available: https://www.microsoft.com/en-us/quantum/

[10] "Google AI Blog: A Preview of Bristlecone, Google's New Quantum Processor," accessed on 2019-07-19. [Online]. Available: https://ai.googleblog.com/2018/03/a-preview-of-bristlecone-googles-new.html

[11] R. R. Schaller, "Moore's law: past, present and future," *IEEE Spectrum*, vol. 34, no. 6, pp. 52–59, 1997.

[12] "Intel Rechisels the Tablet on Moore's Law," accessed on 2019-07-19. [Online]. Available: https://blogs.wsj.com/digits/2015/07/16/intel-rechisels-the-tablet-on-moores-law/

[13] "A New Law to Describe Quantum Computing's Rise?" accessed on 2019-07-19. [Online]. Available: https://www.quantamagazine.org/does-nevens-law-describe-quantum-computings-rise-20190618/

[14] "IBM VP says quantum computer commercialization coming in next 3–5 years," accessed on 2019-07-19. [Online]. Available: https://www.techspot.com/news/80222-ibm-vp-quantum-computer-commercialization-coming-next-3.html

[15] M. Mosca, "Cybersecurity in an era with quantum computers: will we be ready?" *IEEE Security & Privacy*, vol. 16, no. 5, pp. 38–41, 2018.

[16] "What is quantum computing?" accessed on 2019-07-19. [Online]. Available: https://www.cbinsights.com/research/report/quantum-computing/#landscape

[17] L. K. Grover, "A fast quantum mechanical algorithm for database search," *arXiv preprint quant-ph/9605043*, 1996.

[18] V. Mavroeidis, K. Vishi, M. D. Zych, and A. Jøsang, "The impact of quantum computing on present cryptography," *arXiv preprint arXiv:1804.00200*, 2018.

[19] "CRYSTALS: Cryptographic Suite for Algebraic Lattices," accessed on 2019-07-19. [Online]. Available: https://pq-crystals.org

[20] J. Hoffstein, J. Pipher, and J. H. Silverman, "NTRU: A ring-based public key cryptosystem," in *International Algorithmic Number Theory Symposium*. Springer, 1998, pp. 267–288.

[21] L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky, "Lattice signatures and bimodal Gaussians," in *Annual Cryptology Conference*. Springer, 2013, pp. 40–56.

[22] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Conference on the theory and application of cryptographic techniques*. Springer, 1987, pp. 369–378.

[23] G. Alagic, J. Alperin-Sheriff, D. Apon, D. Cooper, Q. Dang, Y.-K. Liu, C. Miller, D. Moody, R. Peralta, R. Perlner, A. Robinson, and D. Smith-Tone, "Status report on the first round of the nist post-quantum cryptography standardization process," National Institute of Standards and Technology, US Department of Commerce, Tech. Rep. NISTIR 8240, January 2019. [Online]. Available: https://doi.org/10.6028/NIST.IR.8240

[24] S. Muppidi, M. O'Brien, and W. Rjaibi, "Wielding a double-edged sword," accessed on 2019-07-19. [Online]. Available: https://www.ibm.com/thought-leadership/institute-business-value/report/quantumsecurity

[25] S. S. Gupta, *Blockchain*. John Wiley & Sons, Inc, 2017.

[26] E. O. Kiktenko, N. O. Pozhar, M. N. Anufriev, A. S. Trushechkin, R. R. Yunusov, Y. V. Kurochkin, A. I. Lvovsky, and A. K. Fedorov, "Quantum-secured blockchain," *Quantum Science and Technology*, vol. 3, no. 3, p. 035004, may 2018. [Online]. Available: https://doi.org/10.1088%2F2058-9565%2Faabc6b

[27] R. Khadka, B. V. Batlajery, A. M. Saeidi, S. Jansen, and J. Hage, "How do professionals perceive legacy systems and software modernization?" in *Proceedings of the 36th International Conference on Software Engineering*. ACM, 2014, pp. 36–47.

[28] "Y2K bug — Definition, Hysteria, & Facts — Britannica.com," accessed on 2019-07-19. [Online]. Available: https://www.britannica.com/technology/Y2K-bug

[29] A. Miranskyy and L. Zhang, "On testing quantum programs," in *Proceedings of the 41st International Conference on Software Engineering: New Ideas and Emerging Results*, ser. ICSE-NIER '19. Piscataway, NJ, USA: IEEE Press, 2019, pp. 57–60. [Online]. Available: https://doi.org/10.1109/ICSE-NIER.2019.00023

[30] L. H. Putnam and W. Myers, "Year 2000 work comes down to the wire," *IEEE Software*, vol. 16, no. 1, pp. 90–96, 1999.

[31] T. J. Shimeall and J. J. McDermott, "Software security in an internet world: An executive summary," *IEEE Software*, vol. 16, no. 4, pp. 58–61, 1999.

[32] J. E. Schultz, "Managing a y2k project-starting now," *IEEE Software*, vol. 15, no. 3, pp. 63–71, 1998.