Literature Survey of Post-Quantum Cryptography

Michael Norberto[1*], Tanis Anderson[1] and Peterling Etienne[1]

[1*]Department, Florida Atlantic University, 777 Glades Rd, Boca Raton, 33431, Florida, US.

*Corresponding author(s). E-mail(s): mnorberto2017@fau.edu; Contributing authors: tanderson2018@fau.edu; petienne2020@fau.edu;

Abstract

Current cryptography methods rely heavily on the how computationally infeasible it is to use brute force attacks on trapdoor functions. In particular, methods like RSA and ECC rely on the difficulty of three algorithms: the Integer Factorization, Discrete Log, and Elliptic-Curve Discrete Logarithm problems. However, recent advances in quantum computing mean that algorithms like Shor's Algorithm, that take advantage of quantum mechanics to quickly and efficiently make accurate guesses on prime factors and solve logarithmic problems, are able to break our current encryption methods in a matter of seconds. To rectify this, newer approaches to encryption are being found and used in what is called "post-quantum cryptography". These post-quantum cryptographic algorithms use many different methods of encryption, including complex polynomial math, stateless hash based approaches, and more. These new forms of encryption can provide the standards of protection against non-quantum attacks that are necessary, while also protecting against quantum attacks.

Keywords: post-quantum, cryptography, encryption, Shor's Algorithm, Grover's Algorithm, quantum-safe, quantum-resistant, NIST

Summary

1 Introduction

We currently use RSA (Rivest Shamir Adleman) and ECC (Elliptic Curve Public-Key Cryptography) for a lot of our encryption methods. These methods are generally what is called public-key cryptography. In a public-key system, there are two keys used, one is called the public-key which is a specific number or formula used for encrypting messages, and the other is called the private key, which is generally an incredibly large prime number. Public-key cryptography works incredibly well because normal computers are not good at factoring prime numbers, which means that obtaining the private key is pointless in terms of computational time, power, and overall resources. For example, on a normal computer, a password comprised of 3763863863761 would take approximately 4 minutes to crack for a given computer, however, as we increase the length we can see that the time that it takes to crack increases exponentially. So if we just use that number two times in a row to get 37638638637613763863863761, then the new time to crack the password becomes 79 million years. Obviously, there is more complexity to passwords than that, but it gives a good idea of how large prime numbers are harder for computers to crack. However, in 1994 a mathematician at MIT named Peter Shor came up with an algorithm that uses a lot of mathematical cleverness to create better guesses for the factors of a prime number. The algorithm works best in quantum systems, especially quantum systems with a higher number of what is referred to as qubits. In a normal computer system, the most basic unit of information is a bit, which represents a logical state where a value is either 1 or 0. Bit is actually an amalgamation of binary integer, and can be thought of as representing either an on or off state, and then using binary code and increasing layers of complexity, we get the computers that we use today. In a quantum system the most basic unit of information is not a bit, but a qubit (short for quantum bit). These qubits do not represent either on or off like a regular bit, but rather both numbers at once. Essentially, Shor's algorithm takes advantage of how qubits function to create a system that efficiently and effectively outputs the highest probability prime factors of a number. What would normally take current computers say 79 million years to crack, with Shor's algorithm it would take quantum computers a few minutes. This is obviously an incredibly big issue, because we use public-key cryptography all of the time, and so much of the internet, hardware, and infrastructure rely on these methods of encryption that would be rendered almost useless. Luckily, we still have time before reaching that point, because according to several estimates, for Shor's algorithm to be effective enough, it would require a system that has around 1300 to 1600 qubits, and currently the most advanced systems are only beginning to near 200 qubits. The point of this project is to help update hardware for the transition period when Shor's algorithm does become a concern to encryption, and to open avenues for future quantum encryption methods in our software and hardware systems.

1.1 Motivations and Contributions

Cryptography is now a crucial component of every single interaction on the internet. It helps regulate and ensure the safety and privacy of everyday exchanges from financial information to medical information to sensitive government data. Secure cryptography is a necessary part of the internet and is used for billions of transactions and by billions of people. However, with the advance of quantum computers, these algorithms will become obsolete, and new algorithms for post-quantum cryptography will be necessary for the protection of user and governmental data. This paper looks at current solutions in the post-quantum cryptography space, and analyzes the necessity of them, as well as some of the challenges in implementing these solutions.

2 Background Information

2.1 Asymmetric Cryptography

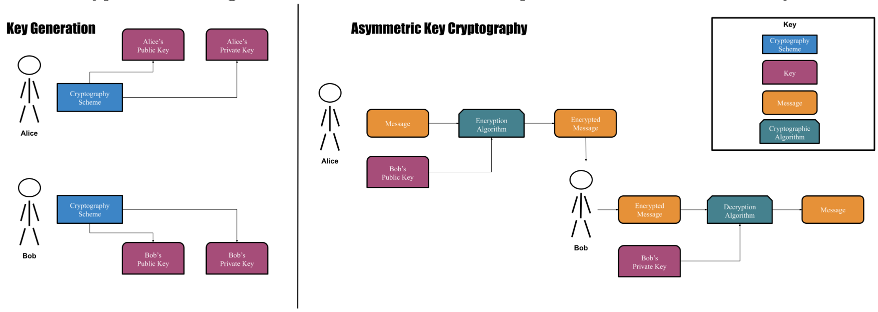
Asymmetric Cryptography is the backbone of our current encryption infrastructure and requires several pieces to function properly. When using asymmetric key cryptography, it can also be referred to as public-key cryptography, this is because the system uses a public key and a private key.

2.1.1 Public Key

The public key in a public-key system is generally a number generated by a cryptographic function for each user in a system. It is known by everyone, and is used to encrypt messages. For example, if Alice, wants to send a message to a different user, Bob, but doesn't want the information to be known to a malicious third party, then she can use Bob's public key, which is known to everyone, to encrypt the message. This does not compromise the security of the message, but means that it is now encrypted, and can in theory only be decrypted by Bob who has the private key that goes with his public key [1].

2.1.2 Private Key

The private key in a public-key system is generally a number, or set of different numbers used in tandem, that correspond to a user's public key. Private keys are what make public-key systems work, they give the user the knowledge needed to take a message encrypted with their public key and decrypt it. Without the private-key it is computationally infeasible to decrypt the message. This is what's called a trapdoor function or a one-way function.



2.1.3 Trapdoor Functions

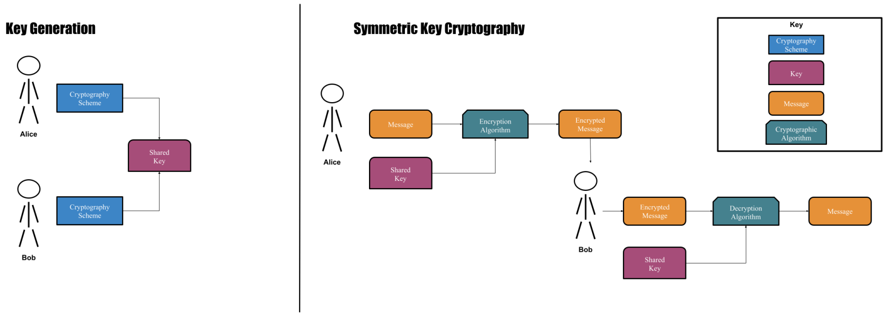
Trapdoor functions provide a major part of the foundation of modern cryptography. These functions are called trapdoor functions or one-way functions, because they work to take a message, perform an operation on it, and then obscure it to the point where it can't be undone. However, in cryptography, these functions have clever mathematics that allows a user with the right knowledge to undo the operation and get back to the original message. It is similar to taking 100 different pages of paper and shredding them. If a third party tries to put them back together, it will take them an unreasonable amount of time, however, since what was on each page is known to us, it is much faster, but still not instant, for us to piece the pages back together [2].

2.1.4 Putting it Together

In a public-key cryptosystem, there are several necessary pieces. There is the system itself, which is needed to generate the public and private keys, to encrypt and decrypt messages using trapdoor functions, and finally a signature scheme is necessary. Signature schemes are used to prove the authenticity of the message, and can be thought of like signatures on check. With all of these pieces, a system is capable of fully encrypting and decrypting messages and ensuring authenticity and message integrity.

2.2 Symmetric Cryptography

Symmetric Cryptography systems have been used in some form throughout history. Symmetric Systems unlike asymmetric systems require that there be a key in some form. That key is then shared between the two participating parties, and is then used for both encryption and decryption. One of the first examples was the Caesar Cipher, which was a basic shift cipher. This cipher used a simple alphabetic shift pattern, where the two participating parties would agree on how much to shift a letter. For example, shifting the alphabet over by one, so A becomes B and B becomes C and so on. This knowledge is required for both the encryption and decryption, and is needed by both parties for the Caesar Cipher, and symmetric systems generally, to be used correctly.



2.3 Quantum Cracking

As mentioned in the Introduction, with the increasing power of quantum computers, the feasibility of attacks against current computational methods is rapidly increasing. Conventional cryptography relies on the difficulty of three problems. The first problem is the Discrete Log Problem, the second is the Integer Factorization Problem, and the final is the Elliptic-Curve Discrete Logarithm Problem. All of these underlie current methods. For example, RSA encryption uses the difficulty of the Integer Factorization Problem to create the keys necessary for encryption and decryption, and ECC uses the difficulty of the Elliptic-Curve Discrete Logarithm Problem. However, Shor's Algorithm, came in part, as a solution to the Discrete Log Problem, and is able to rapidly and efficiently solve all 3 of the aforementioned problems using a quantum computer. This means that the main method of protection for many cryptosystems will be entirely obsolete in the face of operational quantum computers.

2.4 Post-Quantum Algorithms

Post-Quantum Algorithms are the necessary response to the increasing power of quantum computers. Due to the algorithms mentioned above like Shor's Algorithm and Grover's Algorithm, many of the most ubiquitous encryption methods will rapidly become obsolete. Post-Quantum Algorithms are cryptosystems that are specifically designed to be able to withstand both regular attacks and quantum ones. These algorithms generally use more complex mathematics, and focus on new solutions to avoid using prime number mathematics. For example, some systems use a polynomial lattice structure, which essentially limits the possible permutations of the polynomial after each mathematical operation. One example is the NTRUEncrypt algorithm, which uses trapdoor operations on randomly generated invertible polynomials to perform encryption and decryption [3]. Other Post-Quantum Algorithms use modified versions of existing protocools. For example the SIKE algorithm uses Supersingular Isogeny Key mathematics. The algorithm uses a modification of the Diffie-Hellman algorithm, which has been a long-existing algorithm and would allow for easier implementation methods [4]. Other systems may use complex matrix mathematics, or other, newer methods.

3 Literature Survey

Within the past few decades encryption has become one of the most ubiquitous technologies in the world and is used and required for essentially all digital transactions and exchanges. It is a crucial part of the security and privacy that many citizens have come to expect, and it is a crucial part of corporate and governmental data. However, while the necessity and ubiquity of encryption have increased in the past decades, the attacks against them have as well [5] . There is now an increasing threat by quantum computers to current encryption methods as established above. If this threat perseveres then the security and privacy of citizens, corporations, and governments will be at stake. Highly sensitive documents could be exposed, military, traffic, and energy systems could all be open to more frequent and dangerous attacks because of it [6]. The most pressing threat comes from Shor's algorithm [7], a quantum algorithm created by Peter Shor over 20 years ago, that is capable of defeating many current public-key encryption and signature methods in polynomial time, compared to the superpolynomial time that it currently takes to break many of these algorithms. Shor's algorithm presents a somewhat unique challenge because there are no functional implementations of it yet, but the threat alone has been a major impetus behind many of the biggest breakthroughs in cryptography in the past several decades. Organizations like NIST and governments are working to find alternatives to current methods of encryption that are resistant to quantum and classical attacks (NIST Post-Quantum Cryptography Challenge)[8]. These algorithms are the ones called post-quantum algorithms.

3.1 Issues faced

As previously established, the need for post-quantum cryptography is increasing exponentially. To ensure the security of data, infrastructure, hardware systems, and more, it is important to quickly implement and roll out these algorithms. However, as shown by previous implementations of algorithms and their respective roll outs, this is much easier said than done. For example, the roll out of AES is still occurring as of writing this, and the roll out began approximately two decades ago [9]. AES was simply upgrading to a newer version of the same algorithm. However, many of the post-quantum algorithms that are discussed here build on entirely new systems and would likely have an even longer time to roll out and accept. While there are many other problems that post-quantum cryptography must overcome, this is likely the largest. Many of the problems that are faced by post-quantum algorithms are addressed in the proceeding sections, but the roll out still remains the largest hurdle. The change to post-quantum algorithms will be time-consuming and for many companies and organizations, it may be incredibly costly as well, but the alternative of not securing data will likely be much worse. Beyond the roll out and acceptance of post-quantum algorithms, there are many more specific and technical problems that they face.

Another key problem is the computational feasibility and stability of these algorithms. For example, while certain families of algorithms appear to hold great promise, the implementations of them can be incredibly difficult and fragile. This is a problem that other algorithms seek to fix. Another issue faced is that while post-quantum algorithms must be able to withstand quantum attacks, they must also be able to withstand classical attacks. This means that they almost have to fight a war on two fronts. Finally, another problem that these algorithms face is that they must be scalable. It is important that they are able to scale up as needed, and that the parameters or bit sizes currently are enough to meet current standards, but that these algorithms can also deal with larger parameter sizes. This is a key issue for the future when it will be necessary to increase the bit sizes as computers improve.

While there are more problems than just the ones mentioned above, they are covered in the proceeding sections. There is not a single best answer, choice, or algorithm that can easily resolve all of the aforementioned issues, but each algorithm has its own benefits, and each helps to move us closer to a better solution, a more diverse crypto ecosystem, and a more private and secure future.

3.2 Updates on Existing Algorithms

Many algorithms are using some variation of a few larger categories. One such algorithm is the Multivariate Polynomial algorithm. This is a HFEv- signature system that has been unbroken up until at least September 2017 (Bernstein, Daniel). The HFEv- key is a sequence of polynomials in the n-variable ring F[x1,...,xn] over the field F where m ¡= n. These polynomials are quadratics with no squared terms, and each polynomial has the form seen below:

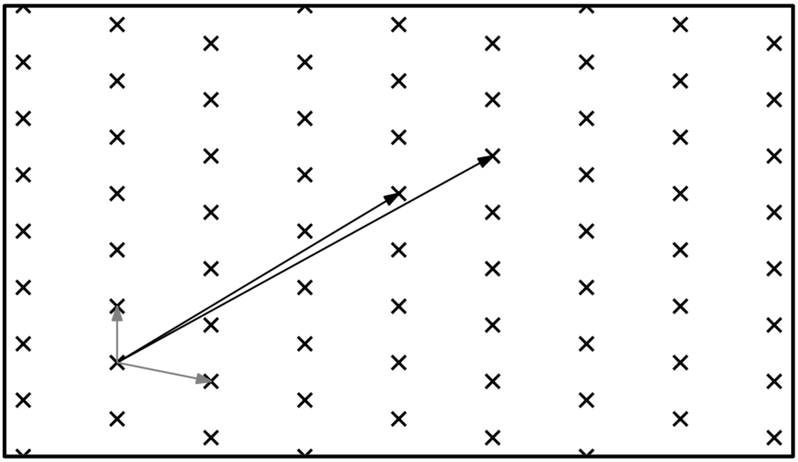
$$a_i + \sum_j b_{i,j}x_j + \sum_{j<k} c_{i,j,k}x_jx_k \text{ with } a_i, b_{i,j}, c_{i,j,k}$$

The signer chooses the polynomials with a certain structure that will allow him to solve these simultaneous quadratics. Thanks to HFEv-, if the equations are in just one variable, there are general methods to solve polynomial equations of degree 'd' over finite fields in (dlogq)O(1) time. The signer views an n-bit signature (Sn1, . . . , Sn) that is a randomly chosen v-bit string in the field F, and v ¡= n-m, along with certain bit elements. This view is secret, meaning that v and q cannot be standardized, and the signature is passed to an invertible n x n matrix chosen by the signer, which is also secret. This creates a linear function of S1, . . . , Sn that is not public.The singer also then views an m-bit hash value along with a random (n-v-m)bit string, which is also not standardized (also passed through a secret matrix). The signing process consists of choosing the random bits mentioned, constructing the hash value, and then trying to solve for S as such: There is a polynomial that specifies the secret equation connecting S and H, P(S, r1,...rv). The signer writes each bit of S as a linear combination of s1, . . . , sn to convert the equation into public quadratic polynomials.

Multivariate polynomials are preferred signatures because they create the shortest signatures. However, some of these schemes have been broken in the past. Solutions to multivariate polynomials are NP hard over any field.

3.3 New Classes of Algorithms

One of the most noticeable trends with the newer algorithms is the use of a lattice-based system. These lattice-based systems use lattices as their foundation, which come from group theory and abstract algebra. Lattices are a set of points in an nth-dimensional space that have periodicity [10].



The benefit of using lattice-based algorithms is that they appear to provide protection against classical and quantum attacks. Many of the NIST Post-Quantum candidates use some version of lattice-based cryptography, although they all have a slight variation, modification, or take on it. These will be analyzed in more depth below. However, it is important to note that while lattice-based cryptography remains secure for now, there has been continuous work to improve attacks against it. The biggest challenge has come from lattice-reduction which is an attack that reduces the size and possibilities of the lattice, and can cripple certain algorithms [11].

3.4 Hardware Encryption

Another growing consideration as shown by NIST and their ongoing competitions is the efficacy and use of encryption and post-quantum encryption algorithms on hardware systems. With the recent growth in Internet of Things (IoT) devices, as well as increases in public facing hardware systems for things like infrastructure, it is important to provide these devices with the proper level of security. NIST handles some of this in their post-quantum encryption challenge, but also hosted another challenge for implementations of lightweight encryption algorithms for hardware systems like FPGA's. This would be a big step towards more secure infrastructure, hardware, and IoT systems. A very noticeable trend in the NIST Post-Quantum Cryptography challenge is the additional focus on hardware implementations and optimizations. A look at each algorithm's website can oftentimes show a new focus on the hardware usage of encryption algorithms. For example, on the SIKE website [4], Hardware implementation

- Rami Elkhatib, Reza Azarderakhsh, and Mehran Mozaffari-Kermani. Accelerated RISC-V for post-quantum SIKE, May 2021.
- Reza Azarderakhsh, Rami El Khatib, Brian Koziel, and Brandon Langenberg. Hardware deployment of hybrid PQC, Apr. 2021.
- Jing Tian, Bo Wu, and Zhongfeng Wang. High-speed FPGA implementation of SIKE based on an ultra-low-latency modular multiplier, Sep. 2020. To appear in IEEE Transactions on Circuits and Systems I (2021).
- Rami Elkhatib, Reza Azarderakhsh and Mehran Mozaffari-Kermani. Efficient and fast hardware architectures for SIKE round 2 on FPGA, May 2020. Published in ARITH 2020.
- Pedro Maat C. Massolino, Patrick Longa, Joost Renes and Lejla Batina. A compact and scalable hardware/software co-design of SIKE, Jan. 2020. Published in TCHES 2020.
- Brian Koziel, A-Bon Ackie, Rami El Khatib, Reza Azarderakhsh, and Mehran Mozaffari-Kermani. SIKE'd Up: Fast and Secure Hardware Architectures for Supersingular Isogeny Key Encapsulation, Jun. 2019. Published in IEEE Transactions on Circuits and Systems I (2020).
- Brian Koziel, Reza Azarderakhsh, and Mehran Mozaffari Kermani. A high-performance and scalable hardware architecture for isogeny-based cryptography, Mar. 2018. Published in IEEE Transactions on Computers (2018).

there are 11 entries on software implementations, and 7 entries on hardware implementation. While there is a discrepancy in the numbers, it helps illustrate the new importance that hardware implementation is being given. The importance of having a secure ecosystem including hardware cannot be overstated, and the NIST Evaluation of Post-Quantum Algorithms on Hardware [12] helps demonstrate this, while also effectively evaluating post-quantum encryption and signature algorithms.

The hardware evaluation breaks down the efficacy of post-quantum algorithms in part by their security level, power requirements, latency, space occupation, and several other factors. This effectively means that there is not necessarily one best algorithm recommended by them, but rather the algorithms are recommended based on their categories or possible uses. For example, they make recommendations on algorithms for the lowest security levels, for the best latency, for the best security, for use on IoT devices, for use on servers, and several other categories. These will be broken down below.

As mentioned above, for hardware systems, there is not a single best post-quantum encryption or signature algorithm. Each one has different uses, and is dependent on the user's needs. One of the key points was that there are tradeoffs for each implementation. For example for lower security applications, CRYSTALS-KYBER still provides good security, and has the lowest latency of any other algorithm, but requires the most computing power. While other algorithms like SPHINCS+ and NTRU-HRSS have higher latencies, but are less costly and would function better for FPGA's or IoT devices. Other algorithms like SABER, provide high security and latency and could be used for server applications where both are necessary. One of the final notes that the paper made was that many of the Key Encapsulation Mechanisms (KEM's), that had higher security levels, could have their latencies reduced by using loop unrolling. For signature algorithms, their latency could be best reduced using loop pipelining.

There are several key takeaways from the NIST Hardware Analysis, and for hardware implementations generally. It is imperative that hardware systems have encryption and signature algorithms on them, especially as they become more interconnected and public facing. While it is important to remember that, it is also important to remember that there is not one best answer, the different algorithms discussed each have their uses and tradeoffs. Specific applications will require specific algorithms to optimize them for respective jobs. It is especially important to remember that there are tradeoffs for increasing the speed and decreasing the latency, and that for algorithms on some smaller devices, it is more important to have security than speed.

3.5 Post-Quantum Signature Algorithms

Another pressing issue facing cryptography and cybersecurity with the advent and improvement of quantum computing is how to handle identification, verification, and authentication. Current methods for digital signatures work well, but Shor's algorithm will be capable of breaking these methods, which could mean spoofing, or replaying attacks. This is why there is currently work being done to build quantum-safe signature systems. However, because there are still not fully functional quantum computers, testing the security in a real-world situation is essentially impossible, and several candidates still need more evaluation for efficacy against classical attacks. As stated by NIST in their evaluation of Post-Quantum Asymmetric Cryptographic Algorithm Candidates [13], there are promising candidates in the field. There do appear to be promising candidates that use some new and updated signature schemes. Although, as NIST points out, the usage of two signature algorithms could be an effective tool for adding in more security than before. They deem this method the "Belt and Suspenders" method, where there are two signature algorithms in use. One is a classical signature algorithm, and the other is a post-quantum signature algorithm. This method would help alleviate concerns about security flaws that each one has, and would allow for slightly smaller bit/encryption sizes for each signature, which could help increase the speed of network-based transactions. NIST also seems to indicate that while there are classical solutions alone that could work with some updates, including Hash Based or Code Based, these would likely be too large, too slow, or not secure enough against quantum algorithms. The solution then seems to be newer algorithms, with NIST suggesting Learning With Errors or Ring Learning With Errors.

While NIST makes those suggestions, there are other candidates that have since emerged that could be better solutions, including algorithms by the name of FALCON, Rainbow, and CRYSTALS-DILITHIUM. FALCON seems to solve some of the speed problems of previous attempts, in specific, it takes some of the work done on the original NTRUSign algorithm, which was meant to go along with the NTRUEncrypt algorithm that is currently a finalist in the NIST competition, and updates it. FALCON (Jeffery Hoffstein Et. Al, FALCON)[14] stands for Fast-Fourier Lattice-based Compact Signatures over NTRU. As the FALCON paper states, one of the problems that is faced by many post-quantum signature algorithms is an increase in signature size, key size, or complexity, all of which can lead to loss of time, and storage space and make transactions and networking cumbersome. Their work builds on some of the work laid out in the documentation and work for the NTRUEncrypt algorithm, and focuses on using lattice-based algorithms to create smaller signature and key sizes while maintaining security. They were able to test the algorithm against existing frameworks including the NTRUSign framework, GPV, and GGH. They kept the hardware the same for all of the testing, and were able to demonstrate that FALCON had substantial performance improvements due to their modifications and methodologies. The team also tested common attacks against lattice-based structures, including lattice reduction, but also signature forging attacks and more, and the algorithm appeared to withstand reasonable versions of these attacks. Additionally, because the algorithm has an operational cost of O(n log n), there is scalability for the algorithm, and it can be used for a long time in the future.

Other Algorithms like the CRYSTALS-DILITHIUM (CD) (Bai Et. Al, CRYSTALS-DILITHIUM) [15] algorithm also take advantage of lattice-based approaches, but try to create somewhat new algorithms based on existing approaches. The CD algorithm also uses a lattice-based framework and focuses, like the NTRU algorithms, on truncated ring mathematics. This effectively means that it should be capable of smaller bit and key sizes, but the same or improved levels of security. CD uses a modification of the SHAKE-256 algorithm with the use of truncated ring mathematics. This provides another layer of security, while also allowing them to have one of the smallest signature and key sizes available. As per NIST's hardware comparison documentation (Basu et. al, NIST Hardware Comparison) [12] CD seems to have some of the lowest latency and or highest speed out of any other signature algorithms, but also appears to require more computational power as it expands to take almost full advantage of the computing power it is provided. One of the key takeaways is therefore that CD appears to provide high security based on the difficulty of Module Learning With Errors problems and the foundational lattice framework, and allows for smaller signature sizes, but at the cost of power. Therefore, for lower powered devices, it is likely a better idea to choose some of the higher latency algorithms like SPHINCS+, which seem to be able to provide good security for lower power costs.

However, a problem that many lattice-based algorithms, and especially truncated ring/polynomial based algorithms run into is that the implementation can often times be difficult and somewhat fragile. That is why other algorithms are using different implementations to create more robust implementations. This is one of the problems that SPHINCS+ (Bernstein et. al, SPHINCS+)[16] tries to rectify. SPHINCS+ uses a stateless hash-based system, which provides several benefits over the previously mentioned methods. In particular, hash-based signatures are well tested and well-known, while providing a high level of security. The stateless approach that SPHINCS uses allows for better post-quantum security, while also keeping some hash sizes down. Stateless hash-based systems have some advantage over regular systems, as they use a few-time signature scheme and builds on the work of Reyzin, Reyzin Better than BiBa: Short One-time Signatures with Fast Signing and Verifying [17]. This foundational work helps demonstrate how SPHINCS+ is able to provide high security while also keeping signature sizes small using a modification of the one-time hashes from a merkle tree or alternative. In their paper, Bernstein et. al, were also able to demonstrate that SPHINCS+ has significant advantages over other post-quantum alternatives like Picnic as it is able to sign 70% faster and verify 50 times faster than Picnic for comparable or better security.

As mentioned above, Picnic is a post-quantum signature algorithm that provides an alternative method set to systems like FALCON, SPHINCS, or CRYSTALS-DILITHIUM. Picnic uses a non-interactive zero-knowledge proof system. Systems like these build on the work of zero-knowledge systems, which are systems that have two participants, a prover and a verifier. The prover issues commitments to the verifier, and then the verifier issues challenges to the prover that they must comply with. In this situation neither party has any prior knowledge or agreements and this is what helps ensure the security. In a non-interactive zero-knowledge proof system (NIZK), the verifier is no longer an interactive second party, but is instead replaced by a randomized hash function or some equivalent (Rackoff et. al, Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack)[18]. The benefit of a NIZK system is that it appears to provide quantum-safe security. The Picnic signature scheme takes advantage of this and works to create a more compact and efficient NIZK system that is optimized to withstand quantum attacks. As mentioned above, in comparison to another signature algorithm like SPHINCS+, it is slower for larger signatures, but an optimized version of Picnic shines with smaller signature sizes and signing times. In summation, although Picnic and other NIZK algorithms cannot outperform hash-based signatures in some metrics, in others like signature sizes and signing times for larger signatures, they can beat them for smaller signature sizes (Kales, Zaverucha, Improving the Performance of the Picnic Signature Scheme)[19]. NIZK systems also seem as though they may be able to provide higher scalability if the efficiency for larger signatures can be increased.

An analysis of post-quantum signature algorithms shows a few interesting trends. There are multiple schemes that can be used to provide post-quantum and classical security for these crucial systems, but each one appears to have its tradeoffs. Overall, CRYSTALS-DILITHIUM appears to provide some of the best security based on the difficulty of the Module Learning with Errors problem. Additionally, based on the NIST hardware review, it appears that CD is the fastest signature scheme available, with its sibling public-key algorithm, CRYSTALS-KYBER, also appearing to be one of the fastest algorithms. However, for embedded and IoT systems, it seems that CD and CK may require and take advantage of too much power. This means that systems like SPHINCS+, FALCON, or Picnic could provide better lightweight alternatives, while still giving post-quantum security. The scalability and longevity of these systems are also important aspects. While CD appears to provide great security now, the continuous reduction in the difficulty of other lattice-based problems could mean that with more time CD will not be nearly as secure, and a similar problem exists for the FALCON signature scheme. Other systems like SPHINCS+ could run into issues of scalability due to its use of hash-based signatures, but there are solutions that can account for this, and the stateless hash-based system that is used gives it more scalability than prior hash-based systems. Finally Picnic appears to have a lot of scalability and potential, but NIZK systems are not well tested enough to make a recommendation yet. Overall though, the winner appears to be CRYSTALS-DILITHIUM, with the best lightweight hardware implementation going to SPHINCS+.

3.6 Grover's Algorithm and Post-Quantum Symmetric Encryption

While we have performed an in-depth analysis of post-quantum asymmetric cryptography in this paper, it is also worth discussing symmetric cryptography. While Shor's algorithm is the main concern for post-quantum asymmetric algorithms, Grover's Quantum Search Algorithm (Grover's Algorithm), is the main concern for symmetric encryption algorithms. Although, the concern for attacks against symmetric encryption is much less, because Grover's algorithm, while effective, as it can essentially halve the effectiveness of the parameters, can be relatively easily countered by increasing the bit sizes and parameters of symmetric algorithms like AES (Grover, A fast quantum mechanical algorithm for database search)[20]. It is especially feasible to do this with AES, as it only uses 128-bit or 256-bit parameters right now, so there is still a lot of room for scalability. Currently, there is not a need for anything beyond that, although there are cryptographers that are already preparing for the inevitable need for post-quantum symmetric encryption algorithms.

The need for post-quantum symmetric algorithms may not be foreseeable for the next several decades at least, but it is still important to begin work on it now. Just because Grover's algorithm is currently the best quantum algorithm to attack symmetric algorithms, does not mean that it will remain as such. If the algorithm is refined or improved, or if a new algorithm is made that is better at solving symmetric encryption problems, then it will be crucial and imperative to have existing post-quantum symmetric algorithms that can be implemented and rolled out to the public.

Advanced Encryption Standard (AES) is a block cipher algorithm that is part of the RIJNDAEL family of algorithms. It undergoes iterative rounds of encryption and manages to be one of the most secure symmetric algorithms out there. As mentioned above, Grover's algorithm is not a true threat to the AES-256 version as of right now, but this could change. This is where the Extended Advanced Encryption Standard (eAES) algorithm comes in. It is a modification and update of the existing AES algorithm, but with more rounds, an updated key scheduler, and an option for a 512-bit version (Bogomolec et. al, Towards Post-Quantum Secure Symmetric Cryptography: A Mathematical Perspective)[21]. The 256-bit version of eAES uses 22 rounds, which is 8 more than for AES, and provides more security than needed to protect against the best classical attack, which can break 11 rounds of encryption. Additionally, replacing the key scheduler, which was previously not cryptographically secure, with cryptographically secure pseudo-random number generators like cSHAKE, means that the algorithm will be even more secure against other classical attacks and against quantum attacks.

Overall, while there is not a massive variety to the current post-quantum symmetric encryption algorithms, the work that has already begun is encouraging, and will likely continue to improve our encryption standards over the coming decades. While simply increasing the bit/parameter sizes for standard AES would provide enough security, using the alternative of eAES will help future-proof against quantum attacks, and will close current avenues that classical attacks use.

4 Comparisons and Gaps in Information

One of the key issues that was faced in writing this paper, is that for each implementation and algorithm, the team behind it tested it using different hardware systems. For example, some used Skylake processors, while others used Xeons. Another issue was that some teams would optimize their own algorithm but would not optimize the competitor they were comparing against. This presents a somewhat unique problem of comparing algorithms that theoretically perform a similar function, but have entirely different benchmarks and benchmarking systems. The solution to this, beyond running benchmarks on our own systems, was more simply, looking at analyses performed by NIST and the National Science Foundation (NSF).

NIST's hardware analysis and comparison has been covered already in the paper, but provides a relatively equitable comparison that uses the same hardware systems, and does its best to optimize each algorithm as best they can, or uses the non-optimized version. This means that the results given are much more equitable than looking at comparisons from each finalists' papers. As previously mentioned, NIST states that for lower level security (level 1), the best algorithm for speed appears to be CRYSTALS-KYBER for public key cryptography, and CRYSTALS-DILITHIUM for signature algorithms. These provide the best speed, but require higher power systems. Other algorithms like SPHINCS+ or NTRU-HRSS have higher latencies, but are more lightweight and would be better suited for implementations on IoT or low power hardware devices. Other algorithms like SABER which provide higher security (level 3), require better power and storage but provide low-latency encryption options, which means that they are better for server applications. One of the key trends that is seen is how different types of algorithms, once optimized, can be faster than others. For example, CRYSTALS-DILITHIUM uses MLWE and RLWE, and can be highly optimized, while other algorithms like SPHINCS+, which use a stateless hash-based system can only be optimized to provide higher speed in certain areas. The Classic McEliece algorithm, which is a code-based algorithm has some of the highest latency of any other encryption or KEM algorithm, but does not provide a fair comparison, as it is tested for the highest security level (level 5), which no other algorithm was tested on. There will always be gaps in the data, but for hardware systems, one of the most important things is tailoring the algorithm to the task at hand. Certain algorithms are much better for specific applications, or for different security levels. However, Classic McEliece was shown to provide the highest security with only minimally higher latency than other algorithms that had the lowest security level, and algorithms like SPHINCS+ are best suited for lightweight hardware signature implementations, while NTRU-HRSS is best suited for lightweight hardware encryption implementations.

The NSF [22] provided a second key set of data, which was comparisons of these implementations on software-hardware co-design systems. This provides a better look at the speed of the algorithms on software systems. One of the biggest problems again was an equitable optimization of each algorithm for the comparison, but another problem is that this paper is from 2019, meaning that some of the algorithms are no longer in the competition. Looking through the comparisons and graphs, algorithms like CRYSTALS-KYBER and New Hope have some of the lowest latencies for both encapsulation and decapsulation (for KEMs), while other algorithms like FrodoKEM, have incredibly high latencies while providing approximately the same level of security as other algorithms. This again is in part due to the different types of algorithm used, as well as the mathematics behind them. For example, while FrodoKEM and NTRU-HRSS are both lattice-based, NTRU-HRSS uses optimized Fast Fourier Transforms, and simpler mathematics overall. This helps demonstrate again that each algorithm has limitations and can be better suited for certain implementations over others.

It is hard to say if there is one best algorithm, because there isn't. Generally, each algorithm has a task that it could best be suited for, but there are some key algorithms that appear to have promise in terms of security, speed, or overhead. SPHINCS+ for signature algorithms appears to work well for lightweight signature systems. Classic McEliece appears to provide some of the most stable and secure encryption, and SABER appears to be a good compromise between speed and security for hardware. For software-hardware co-design systems, SABER still appears to provide a good compromise, while algorithms like New Hope and NTRU-HRSS had much lower latencies than other Algorithms like FrodoKEM. The reasons as to why will be covered below in the technical discussion.

5 Charts

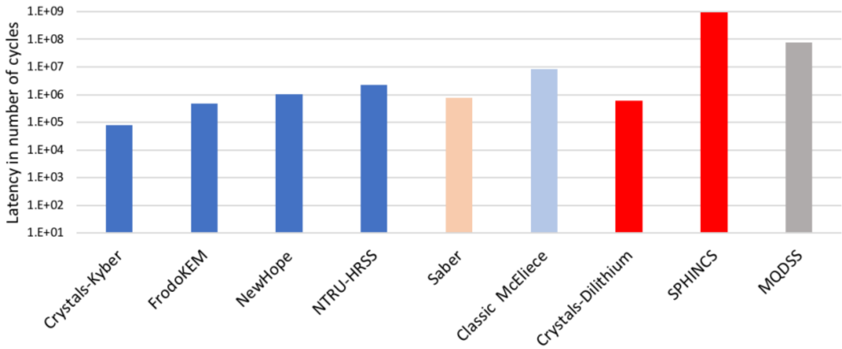


Fig. 1: NIST's Hardware Comparison Chart

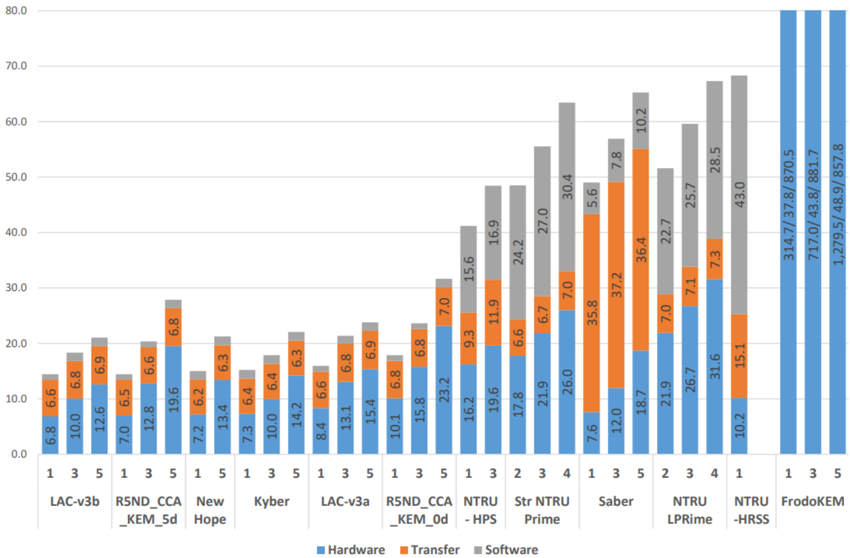


Fig. 2: NSF Encapsulation Software/Hardware Co-Design Chart

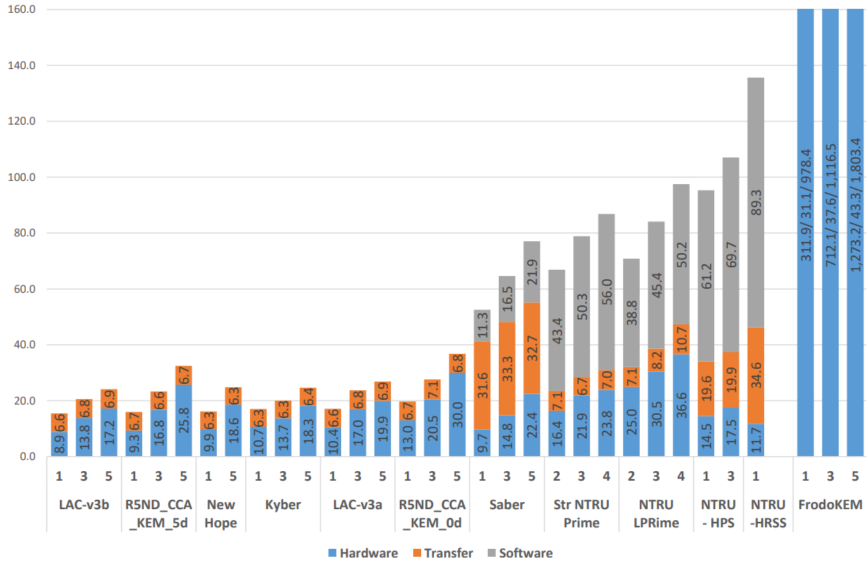


Fig. 3: NSF Decapsulation Software/Hardware Co-Design Chart

6 Technical Discussion

7 Conclusion

In this paper we have examined the threat that functional quantum computers create to our current cybersecurity ecosystem, we have provided background information, we have analyzed different types of algorithms, the eventual need for post-quantum symmetric encryption, and we have compared different algorithms. There is without a doubt a looming threat from quantum computers and algorithms like Shor's and Grover's algorithms. Cryptography relies on the hardness of problems, and Shor's and Grover's algorithms are capable of solving many of our current problems in record time. While it will not be an overnight loss of security and privacy, or even any real loss of security or privacy for an average person, ensuring the safety, security, and privacy of all technology is a necessity for modern interactions. As the interconnectedness of the world increases, as IoT systems become more and more integrated, as infrastructure becomes internet enabled, and as more transactions and data are stored online, the need for security will only increase, but so will the attacks on data and encryption.

As demonstrated in this paper, there are real and functional alternatives to current encryption standards, and while there are problems that each alternative faces, the biggest hurdle for all of them will be rolling them out to the public. This is likely where the most time, money, and energy will be spent. To help ensure the speed and efficiency of this, it is important to select the right algorithm that is capable of providing the right amount of security, scalability, and speed, while also allowing for easy roll out and acceptance. This is where real discussion is required, and while we discuss many of the current candidates, as demonstrated by NIST and NSF themselves, there is not one algorithm that can do everything that is needed. We will need a more diverse cryptographic ecosystem, with algorithms tailored to each task. There will need to be algorithms for lightweight hardware systems, for servers, for internet transactions, for users, for signatures, and more. Selecting the best one out of each of those categories is a slightly easier task, but one that is still incredibly difficult overall.

Based on the data from NIST, for the best security overall, the algorithm appears to be the Classic McEliece algorithm, which has stood the test of time, and provides the highest level of security, while only sacrificing minimal speed in comparison to other algorithms. For the best signature algorithm in the finalist pool, it appears to be CRYSTALS-DILITHIUM, which provides the best speed, and should provide some of the best security based on the hardness of MLWE and RLWE.

It is also worth discussing post-quantum symmetric encryption, which is not yet a necessity, nor will it be for the foreseeable future, but it is worth keeping in mind, as algorithms like eAES provide higher security alternatives to AES, while still remaining similar enough for a relatively easy switch over.

With all of this said though, the biggest takeaways are that there is a dire need for post-quantum cryptography, even if the worst fears about breaking encryption are never realized, having higher security algorithms will help ensure the safety, privacy, and security of citizens, organizations, governments, and their data for decades to come. There is not a single algorithm that can do everything, but there are algorithms that can be tailored to each problem, and using that knowledge, the roll out to newer standards can be expedited. Regardless of if Shor's and Grover's algorithms actually end up impacting average people's security and privacy, post-quantum cryptography is a necessity.

References

[1] Diffie, W.: The First Ten Years of Public-Key Cryptography. http://www.dragonwins.com/domains/getteched/bbc/literature/Diffie88.pdf

[2] Yao, A.C.: Theory and Application of Trapdoor Functions. https://www.di.ens.fr/users/phan/secuproofs/yao82.pdf

[3] et. al, J.H.: NTRU. https://ntru.org/f/ntru-20190330.pdf

[4] et. al, D.J.: Supersingular Isogeny Key Encapsulation. https://sike.org/files/SIDH-spec.pdf

[5] Boneh, D.: Twenty Years of Attacks on the RSA Cryptosystem

[6] Sanger, D.E.: Pipeline Attack Yields Urgent Lessons About US Cybersecurity. https://www.nytimes.com/2021/05/14/us/politics/pipeline-hack.html

[7] Shor, P.W.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. https://arxiv.org/pdf/quant-ph/9508027.pdf

[8] Post-Quantum Cryptography. https://csrc.nist.gov/projects/post-quantum-cryptography

[9] Bos, J.: POST-QUANTUM CRYPTO: WHAT YOU NEED TO KNOW

[10] Micciancio, D., Regev, O.: Lattice-based Cryptography

[11] Langlois, A., Stehlé, D.: Worst-Case to Average-Case Reductions for Module Lattices. https://eprint.iacr.org/2012/090.pdf

[12] Kanad Basu, M.N. Deepraj Soni, Karri, R.: NIST Post-Quantum CryptographyA Hardware Evaluation Study. https://eprint.iacr.org/2019/047.pdf

[13] Tolga Acar, C.C. Josh Benaloh, Shumow, D.: Evaluating Post-Quantum Asymmetric Cryptographic Algorithm Candidates. https://csrc.nist.gov/csrc/media/events/workshop-on-cybersecurity-in-a-post-quantum-world/documents/presentations/session7-shumow-dan.pdf

[14] Pierre-Alain Fouque, P.K.V.L.T.P.T.P.T.R.G.S.W.W. Jeffrey Hoffstein, Zhang, Z.: Falcon: Fast-Fourier Lattice-based Compact Signatures over NTRU. https://falcon-sign.info/falcon.pdf

[15] Shi Bai, E.K.T.L.V.L.P.S.G.S. Léo Ducas, Stehlé, D.: CRYSTALS-Dilithium. https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf

[16] et. al, D.J.B.: The SPHINCS+ Signature Framework. https://sphincs.org/data/sphincs+-paper.pdf

[17] Reyzin, L.R.N.: Better than BiBa: Short One-time Signatures with Fast Signing and Verifying

[18] et. al, C.R.: Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. https://link.springer.com/content/pdf/10.1007/3-540-46766-1_35.pdf

[19] et. al, D.K.: Improving the Performance of the Picnic Signature Scheme. https://github.com/microsoft/Picnic/blob/master/spec/picnic3-eprint.pdf

[20] Grover, L.K.: A Fast Quantum Mechanical Algorithm for Database Search. https://arxiv.org/pdf/quant-ph/9605043.pdf

[21] et. al, X.B.: Towards Post-Quantum Secure Symmetric Cryptography: A Mathematical Perspective. https://eprint.iacr.org/2019/1208.pdf

[22] et. al, V.B.D.: Implementation and Benchmarking of Round 2 Candidates in the NIST Post-Quantum Cryptography Standardization Process Using Hardware and Software/Hardware Co-design Approaches. https://par.nsf.gov/servlets/purl/10175000