Array 15 (2022) 100242

Contents lists available at ScienceDirect

Array

journal homepage: www.sciencedirect.com/journal/array

Post-quantum cryptography Algorithm's standardization and performance analysis

Manish Kumar

Department of Master of Computer Applications, M S Ramaiah Institute of Technology, Bangalore, 54, India

ARTICLE INFO

Keywords:
Quantum cryptography
Code based cryptography
Multivariate quadratic equations based cryptography
Hash-based cryptography
Isogeny based cryptography
Lattice-based cryptography

ABSTRACT

-Quantum computer is no longer a hypothetical idea. It is the world's most important technology and there is a race among countries to get supremacy in quantum technology. It is the technology that will reduce the computing time from years to hours or even minutes. The power of quantum computing will be a great support for the scientific community. However, it raises serious threats to cybersecurity. Theoretically, all the cryptography algorithms are vulnerable to attack. The practical quantum computers, when available with millions of qubits capacity, will be able to break nearly all modern public-key cryptographic systems. Before the quantum computers arrive with sufficient 'qubit' capacity, we must be ready with quantum-safe cryptographic algorithms, tools, techniques, and deployment strategies to protect the ICT infrastructure. This paper discusses in detail the global effort for the design, development, and standardization of various quantum-safe cryptography algorithms along with the performance analysis of some of the potential quantum-safe algorithms. Most quantum-safe algorithms need more CPU cycles, higher runtime memory, and a large key size. The objective of the paper is to analyze the feasibility of the various quantum-safe cryptography algorithms.

1. Introduction

Cryptography is one of the most important pillars of cybersecurity. Cryptography is an art of rendering techniques and the science of securing the information, where the information can be interpreted only by the sender and intended recipient. Most modern cryptography algorithms are based on complex mathematical functions. The design of cryptographic algorithms is based on assumptions of computational difficulty. These are asymmetrical problems, such as integer factorization: multiplying two integers is straightforward, but factoring a 1000-digit integer is difficult.

A form of cryptography that uses two related keys, a public key, and a private key; the two keys have the property that, given the public key, it is computationally infeasible to derive the private key. Current public-key encryption relies on the fact that a classical computer can easily multiply large prime numbers but requires thousands of years to reverse such a calculation. Quantum computing will accelerate the ability to decrypt information protected by current public-key encryption techniques.

Quantum computing is a technology based on the principles of quantum theory which is much faster than classical computing techniques. The classical cryptography algorithms are completely vulnerable to quantum computers. Transitioning to the quantum computer will make the existing IT infrastructure completely unsafe and it requires the design and development of quantum-safe or quantum-resistant cryptography algorithms. The objective of this paper is to discuss various cryptography techniques and potential quantum-safe cryptography algorithms along with the performance analysis of the algorithms.

The paper is divided into eight sections. Section 2 gives general information about cryptography followed by Section 3 introducing quantum computers. Section 4 discusses the post-quantum encryption problem. Many countries and professional organizations are putting effort to standardize the quantum-safe algorithms which are discussed in Section 5. The National Institute of Standards and Technology (NIST) has initiated an intensive process to standardize the post-quantum encryption algorithm. Some of the potential quantum-safe algorithms which are announced by the NIST in 3rd round of the selection process have been discussed in Section 6 along with the performance analysis of the algorithms based on the Open Quantum Safe (OQS) project implementation. The overall observation, implementation feasibility, and migration challenges are discussed in Section 7 followed by a concluding remark in Section 8.

E-mail address: manishkumarjsr@yahoo.com.

https://doi.org/10.1016/j.array.2022.100242
Received 2 April 2022; Received in revised form 4 August 2022; Accepted 4 August 2022
Available online 18 August 2022

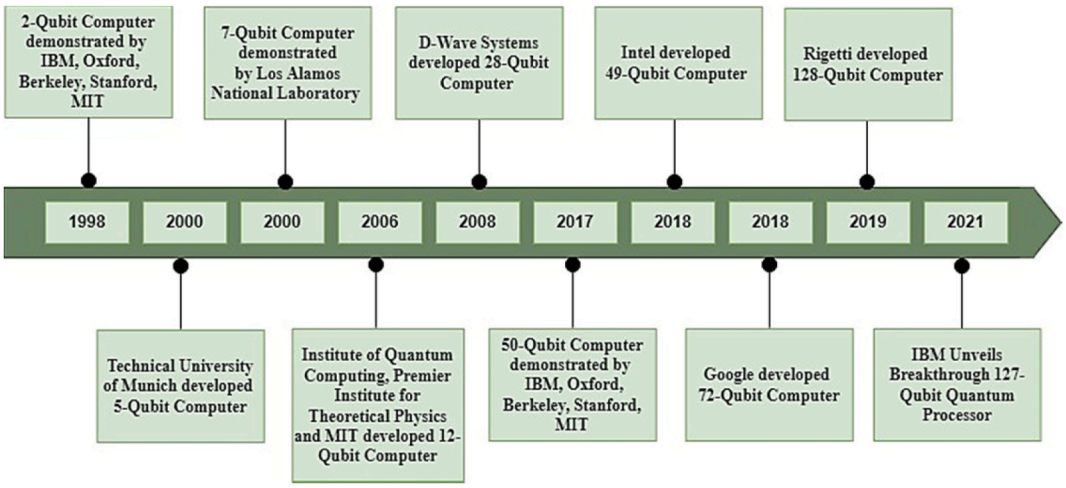

Fig. 1. Quantum computing growth in qubits between 1998 and 2021.

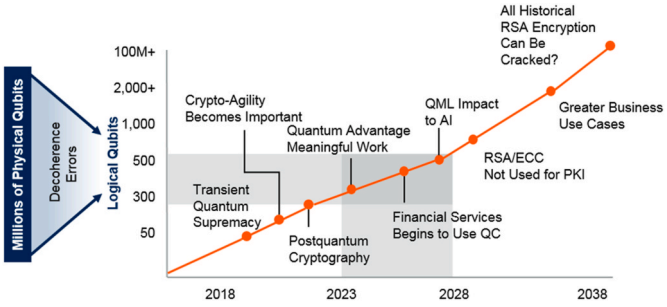

Fig. 2. Qubit Timeline Estimate (Source Gartner [83]).

2. Cryptography –the building blocks of information security

Today, information security without cryptography is unimaginable thought. Cryptography is used in various ways to protect the confidentiality and integrity of the information. Information security hardware and software products implement cryptography in many creative ways with the sole purpose of providing security. Cryptographic algorithms are well-defined procedures or sequences of rules or steps, or a series of mathematical equations used to describe cryptographic processes such as encryption/decryption, key generation, authentication, signatures, etc. [31].

Cryptographic algorithms are broadly classified into two categories i. e., Symmetric Cryptographic algorithms and Asymmetric Cryptographic Algorithms.

i) **Symmetric Cryptography Algorithms:** In symmetric cryptography algorithms, the sender and receiver use the same key to encrypt and decrypt the data. Symmetric key algorithms are very efficient and easy to implement. However, key management in symmetric cryptography is a challenging issue. The key must be shared among the two parties before initiating the communication. Symmetric ciphers are mainly susceptible to chosen-plaintext attack, known-plaintext attack, linear cryptanalysis, and differential cryptanalysis.

ii) **Asymmetric Cryptography Algorithm:** In an asymmetric cryptography algorithm, the sender and receiver use a pair of keys. One of the keys is kept secret i.e., called a private key and the other key is made public called the public key. An asymmetric cryptography algorithm is also known as a public-key algorithm. The sender encrypts the data using the public key and the receiver decrypts the data using

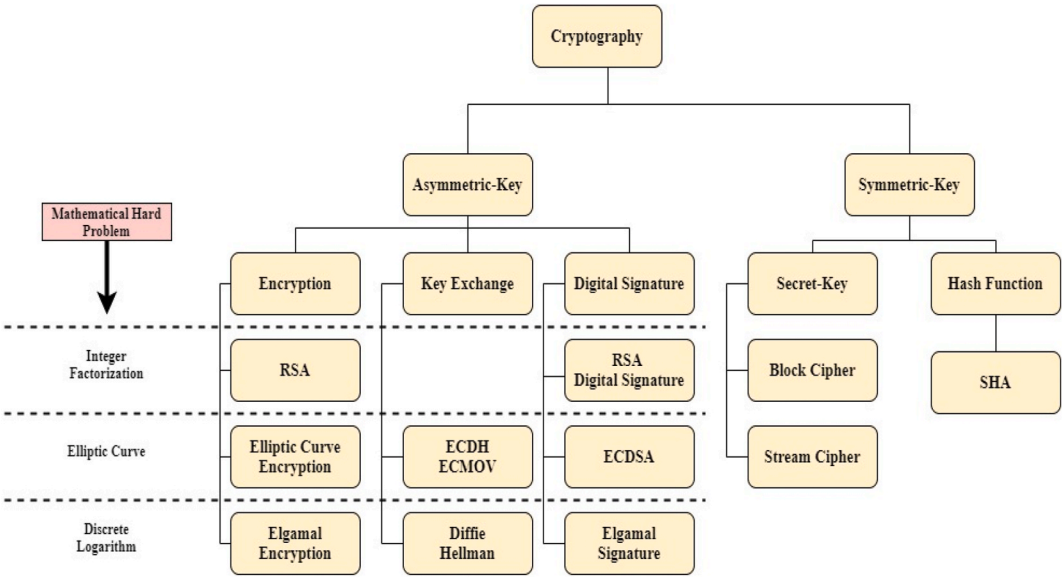

Fig. 3. Cryptography branches and the associated mathematical problems.

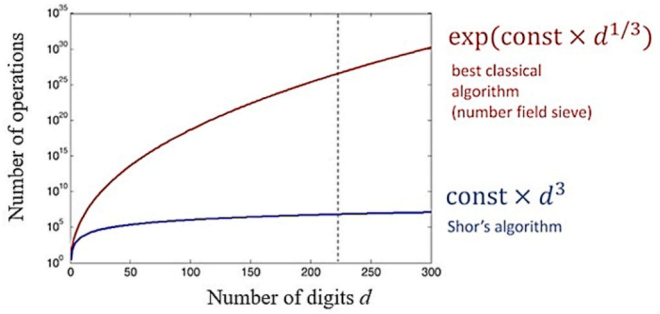

Fig. 4. Performance of classical vs. Quantum algorithms for prime number factorization.

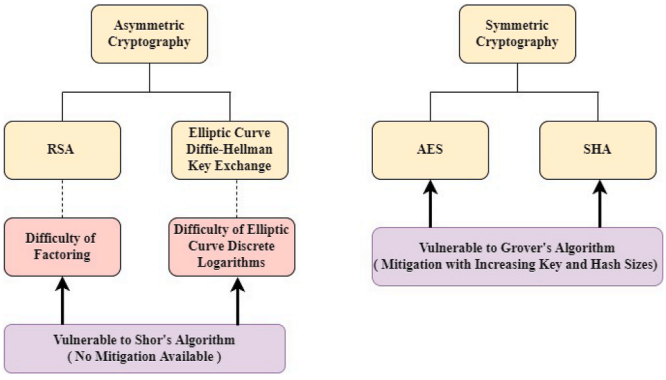

Fig. 5. Effect of Shor's and Grover's algorithms on Cryptography Algorithm.

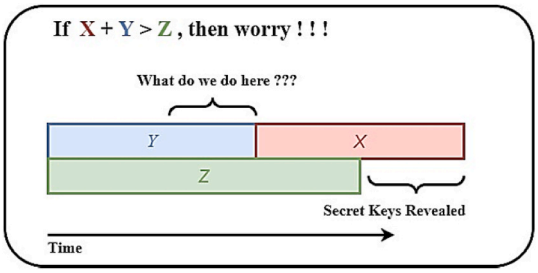

Fig. 6. Theorem (Mosca).

Table 1
Impact of quantum computing on common cryptographic algorithms.

Cryptographic Algorithm	Type	Purpose	Impact from Large Scale Quantum Computer
AES	Symmetric Key	Encryption	Larger key sizes needed
SHA-2, SHA-3	————	Hash Functions	Larger output needed
RSA	Public Key	Signatures, Key establishment	No longer secure
ECDSA, ECDH (Elliptical Curve Cryptography)	Public Key	Signatures, Key Exchange	No longer secure
DSA (Finite Field Cryptography)	Public Key	Signatures, Key Exchange	No longer secure

a private key. Generally, asymmetric cryptography algorithms are resource-intensive and require more processing time and power. Asymmetric cryptography algorithms satisfy the requirements for digital signatures, cryptographic key establishment, and many more security services. The public-key ciphers are mainly vulnerable to chosen-ciphertext attacks, man-in-the-middle attacks, and some specific attacks related to the underlying hard problems.

Symmetric and asymmetric, both the algorithms are vulnerable to brute force attack [35,75]. In a brute force attack, the attacker tries all the possible keys. The resistance of the algorithms depends on the key size and computational power required to generate and apply all the possible keys to break the encryption [22,81,84].

Cryptography does not only secure the data but is also used for confidentiality, authentication, integrity, and accessibility. There are various implementations of cryptography algorithms in our daily life. For example, the use of SSL/TLS to provide secure internet browsing, digital signature for authentication and validation, electronic money transactions, secure data storage, etc. [26,55,58].

The concept of a "computationally secure scheme" is that it is theoretically possible to crack such a system, but practically impossible to do so. The balance is simple to comprehend: if breaking a system costs more than the value it protects, then doing so is pointless. The strength of any cryptography algorithm depends on the computational time and effort required to break the algorithm. Cryptanalysis and brute force techniques are commonly used for breaking the cryptography algorithm. Cryptanalysis is the science of breaking cryptography using mathematics and algorithms. Brute force means that you enumerate overall possibilities until you find the solution you were searching for. The cryptography algorithm assumes to be secure if it is proven mathematically that to break the algorithm, requires huge computational resources and time which is practically not feasible.

There is continuous growth in computational power and accordingly, the cryptography algorithms are also modified to maintain the strength. Most of the cryptography algorithms used today are resilient against a classical computer. A Quantum computer is new computation technology that is completely different than a classical computer. A Quantum computer is much faster than a classical computer.

Quantum computing is the next revolutionary technology. It is the technology that will have the ability to reduce the computing time from years to hours or even minutes. The power of quantum computing will be a great support for the scientific community. However, it raises serious cybersecurity threats. Most of the existing cryptography algorithms are theoretically vulnerable to quantum computing. A practical quantum computer with sufficient numbers of qubits will be able to break all the modern public-key cryptographic systems [16].

3. Quantum computing

A Quantum computer is no longer a hypothetical idea. As quoted by many experts, it is the world's most important technology and there is a race among countries to get supremacy in quantum technology with a quantum computer of a sufficient number of qubits and fault tolerance. US, China, France, the UK, Germany, and Russia are front runners in the race, whereas other countries are putting their best effort to join the league. Race to hold control over "Quantum Computing" technology is not only limited to country, it is significantly driven by the top technology giant like Microsoft, IBM, Google, D-Wave, Toshiba, etc.

Quantum computing got popular after the publication of the article "Simulating Physics with Computer" by American theoretical physicist Feynman [41]. In the article, Feynman suggested the use of quantum states for calculations.

Quantum computing is an application of a quantum mechanism that uses a quantum phenomenon to perform computation. A Quantum computer is a device that performs quantum computing. It manipulates the states of qubits in a controlled way to perform algorithms.

A qubit (or quantum bit) is the quantum-mechanical analogue of a classical bit. In classical computers information is encoded in a bit, where each bit can be either zero or one. In quantum computing, the information is encoded in qubits. The state of the qubits is written as |

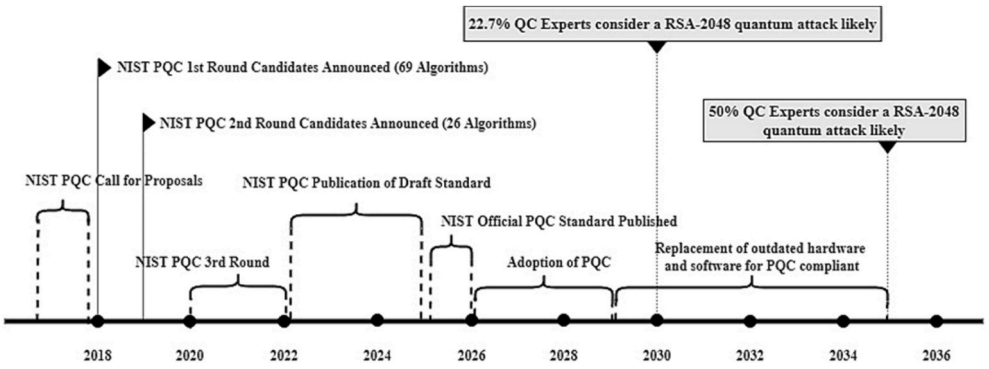

Fig. 7. Plausible Timeline for NIST PQC standards development and adoption.

Table 2
Code-based quantum-safe cryptographic algorithms.

Sl. No.	Algorithm Name	Current Status	Open-Source C Library for Quantum-Safe Cryptographic Algorithms (liboqs)				Usage
1	Classic McEliece	Finalists	NIST Level	Algorithms Name	Public Key Size (bytes)	Private Key Size (bytes)	
			1	Classic-McEliece-348,864	261,120	6452	Key Encapsulation Mechanisms
				Classic-McEliece-348864f	261,120	6452	
			3	Classic-McEliece-460,896	524,160	13,568	
				Classic-McEliece-460896f	524,160	13,568	
			5	Classic-McEliece-6688,128	1,044,992	13,892	
				Classic-McEliece-6688128f	1,044,992	13,892	
				Classic-McEliece-6960,119	1,047,319	13,908	
				Classic-McEliece-6960119f	1,047,319	13,908	
				Classic-McEliece-8192,128	1,357,824	14,080	
				Classic-McEliece-8192128f	1,357,824	14,080	
2	BIKE	Alternates	1	BIKE-L1	1541	5223	Key Encapsulation Mechanisms
			3	BIKE-L3	3083	10,105	
3	HQC	Alternates	1	HQC-128	2249	2289	Key Encapsulation Mechanisms
			3	HQC-192	4522	4562	
			5	HQC-256	7245	7285	

Table 3
Runtime analysis of open quantum safe code based cryptographic algorithms (Key encapsulation mechanisms).

Algorithm	Keygen/s	Keygen (cycles)	Encaps/s	Encaps (cycles)	Decaps/s	Decaps (cycles)
BIKE-L1 (x86_64)	4256.67	587,177	29,602	84,291	1866.67	1,339,145
BIKE-L3 (x86_64)	1345.33	1,858,211	14276.33	174,889	558	4,480,077
Classic-McEliece-348,864 (x86_64)	7.2	347,293,159	55104.67	45,242	18448.67	135,429
Classic-McEliece-348864f (x86_64)	9.24	270,597,954	54,883	45,426	17712.67	141,044
Classic-McEliece-460,896 (x86_64)	2.42	1,032,268,240	31683.67	78,770	7228.33	345,779
Classic-McEliece-460896f (x86_64)	2.91	858,376,391	31720.33	78,686	7220.33	346,146
Classic-McEliece-6688,128 (x86_64)	1.56	1,604,951,815	17,170	145,425	6163.67	405,515
Classic-McEliece-6688128f (x86_64)	2.2	1,137,319,229	17034.67	146,612	6112.67	408,895
Classic-McEliece-6960,119 (x86_64)	1.52	1,647,315,912	17755.33	140,662	6705.67	372,718
Classic-McEliece-6960119f (x86_64)	2.36	1,058,000,162	17,750	140,696	6655.33	375,498
Classic-McEliece-8192,128 (x86_64)	1.55	1,617,672,860	14259.33	175,108	6128	407,881
Classic-McEliece-8192128f (x86_64)	2.13	1,176,481,909	14550.33	171,631	6146.67	406,618
HQC-128 (x86_64)	16412.33	152,121	9847.67	253,741	5710.67	437,665
HQC-192 (x86_64)	7453.33	335,160	4283	583,564	2422.33	1,032,052
HQC-256 (x86_64)	4232.67	590,283	2430	1,028,703	1435.33	1,741,750

$0\rangle$ and $|1\rangle$. The qubits can be simultaneously both at 0 and 1. Quantum mechanism is a strange phenomenon. Quantum computers are built using the following features of quantum states:

i) **Superposition:** Quantum systems can exist in two states at once. A qubit can be in 0 and 1 at the same time. When the measurement is performed, the qubit collapses to either zero or one.

ii) **Entanglement:** It is a quantum mechanical phenomenon where the state of entangled particles can be described with reference to each other. Measurement performed on one entangled particle will immediately influence the other entangled particle irrespective of the distance among the entangled particle.

iii) **Interference:** The fundamental idea in quantum computing is to control the probability of qubits collapsing into a particular measurement state. Quantum interference, a by-product of

Table 4
Memory Use Analysis (bytes per algorithm) of Open Quantum Safe Code Based Cryptographic Algorithms (Key Encapsulation Mechanisms).

Algorithm	Keygen (maxHeap)	Keygen (maxStack)	Encaps (maxHeap)	Encaps (maxStack)	Decaps (maxHeap)	Decaps (maxStack)
BIKE-L1 (x86_64)	11,420	90,552	12,545	25,720	12,577	73,464
BIKE-L3 (x86_64)	17,844	179,448	20,511	50,296	20,543	144,952
Classic-McEliece-348,864 (x86_64)	271,748	2,205,032	276,556	3824	271,940	43,232
Classic-McEliece-348864f (x86_64)	271,748	2,205,032	276,556	3824	271,940	43,232
Classic-McEliece-460,896 (x86_64)	541,904	4,768,360	546,772	6528	542,156	80,256
Classic-McEliece-460896f (x86_64)	541,904	4,768,360	546,772	6528	542,156	80,256
Classic-McEliece-6688,128 (x86_64)	1,063,060	4,768,840	1,067,980	6320	1,063,364	94,144
Classic-McEliece-6688128f (x86_64)	1,063,060	4,768,840	1,067,980	6320	1,063,364	94,144
Classic-McEliece-6960,119 (x86_64)	1,065,403	4,768,808	1,070,309	6224	1,065,693	80,896
Classic-McEliece-6960119f (x86_64)	1,065,403	4,768,808	1,070,309	6224	1,065,693	80,896
Classic-McEliece-8192,128 (x86_64)	1,376,080	4,769,000	1,376,080	384	1,376,384	133,408
Classic-McEliece-8192128f (x86_64)	1,376,080	4,769,000	1,376,080	384	1,376,384	133,408
HQC-128 (x86_64)	8714	40,120	13,259	53,720	13,323	62,872
HQC-192 (x86_64)	13,260	79,640	22,350	107,000	22,414	125,304
HQC-256 (x86_64)	18,706	182,040	33,239	226,200	33,303	255,704

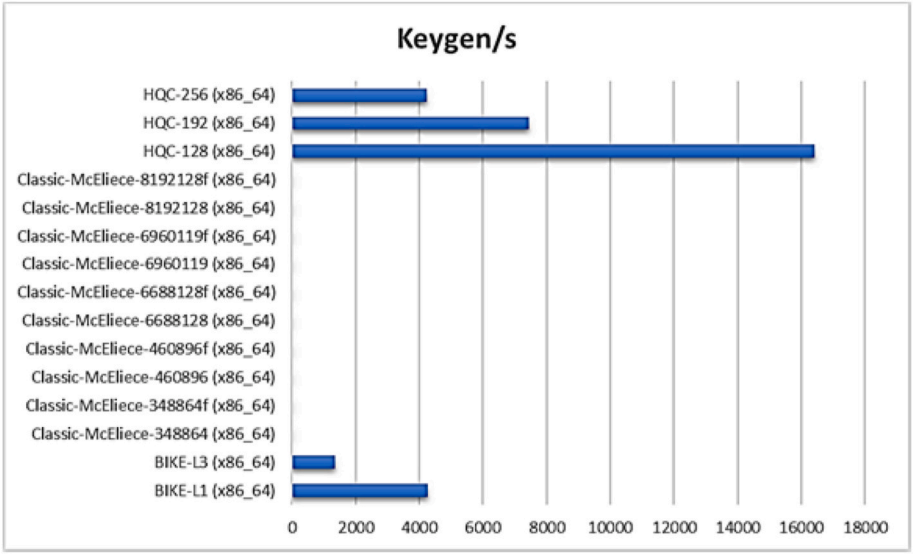

Fig. 8. Keygen operations per second per algorithm (Key Encapsulation Mechanisms using Code Based Algorithms).

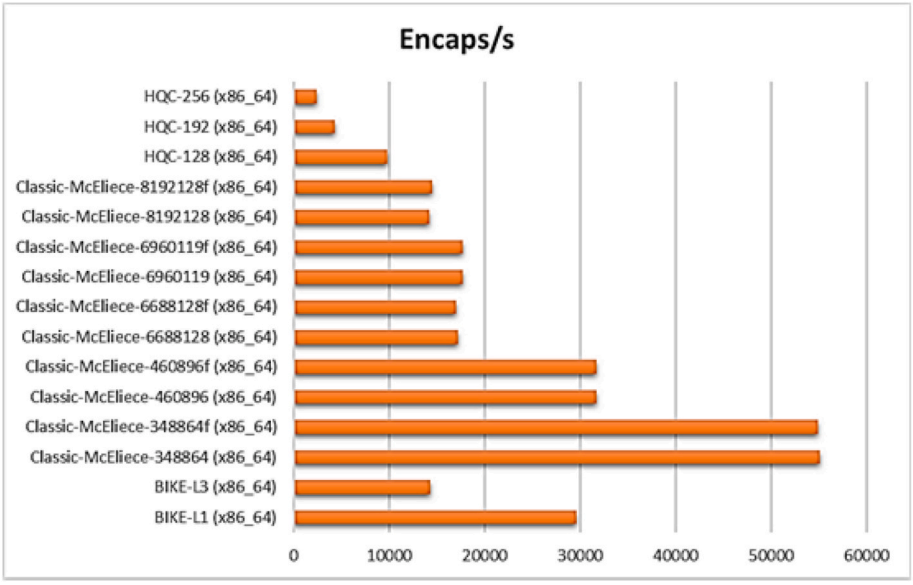

Fig. 9. Encaps operations per second per algorithm (Key Encapsulation Mechanisms using Code Based Algorithms).

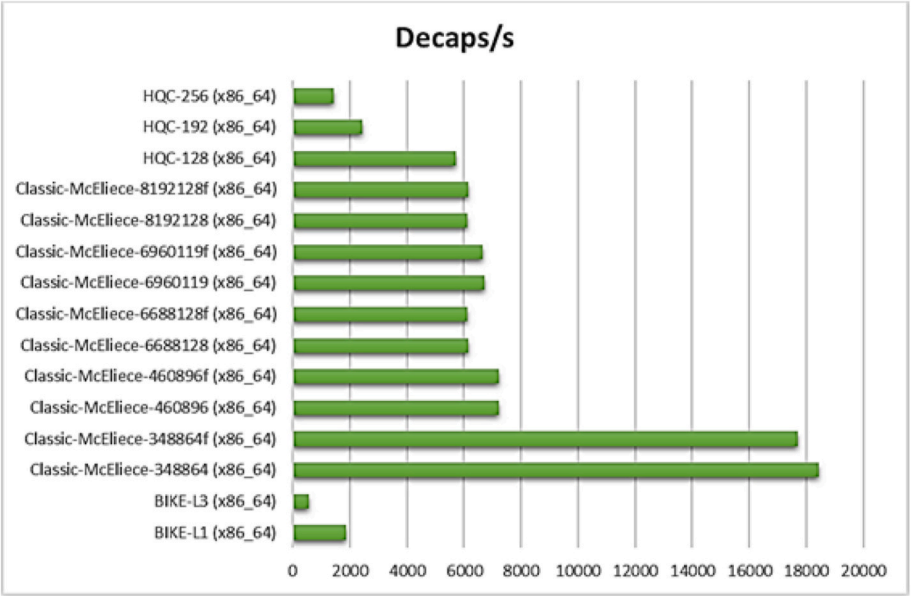

Fig. 10. Decaps operations per second per algorithm (Key Encapsulation Mechanisms using Code Based Algorithms).

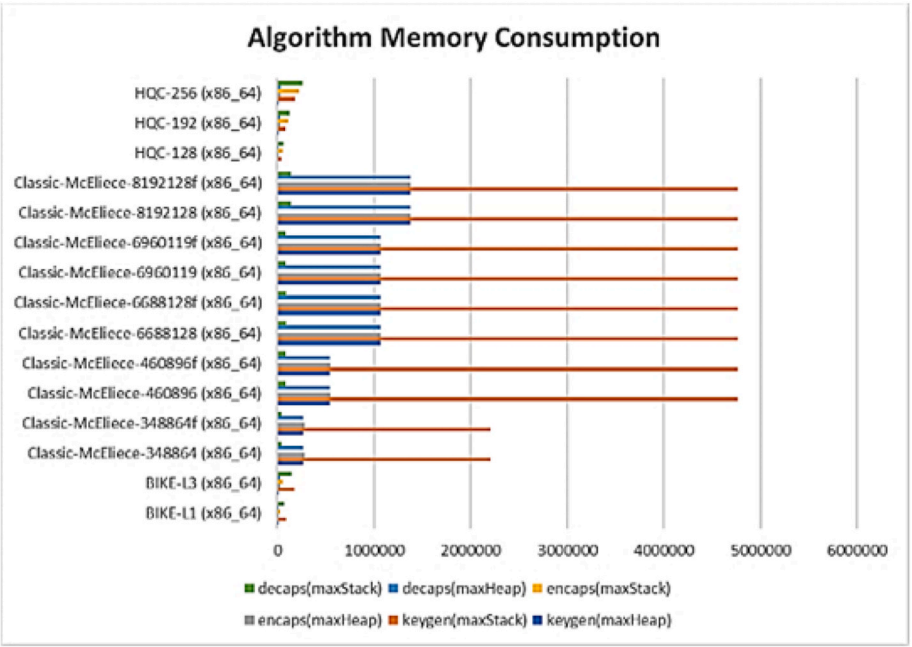

Fig. 11. Memory consumption (bytes per algorithm)(Key Encapsulation Mechanisms using Code Based Algorithms).

superposition, allows controlling the measurement of a qubit toward a desired state or set of states.

3.1. The timescale of quantum computing development

The quantum computing field is growing rapidly. Globally there is exponential growth in this field. Building a quantum computer with higher qubits and precise error control is the sole objective of the researchers. There are remarkable achievements in the last 22 years in this field (Fig. 1). MIT, Oxford, Berkely, and IBM could develop a 2-qubit quantum computer early in 1998. Google developed a 72-qubit quantum computer in 2018. Rigetti in 2019, announced to develop a 128-qubit quantum computer within a year [45,52,57].

The development of the Quantum Computer can be divided into three generation.

i) **First Generation:** The first-generation quantum computers are developed at the early stage for non-commercial use. These models were built for proof of concept with low to medium complexity.
ii) **Second Generation:** Many organizations who got the breakthrough in their initial research and possessed the necessary hardware infrastructure could develop the quantum computer with a higher number of qubits and complexity. The second-generation quantum computers are solely designed and developed for commercial applications and high-end research focused on improved scalability and speed. These quantum computers

Table 5
Multivariate quadratic equations based quantum-safe cryptographic algorithms.

Sl. No.	Algorithm Name	Current Status	Open-Source C Library for Quantum-Safe Cryptographic Algorithms (liboqs)				Usage
			NIST Level	Algorithms Name	Public Key Size (bytes)	Private Key Size (bytes)	
1	Rainbow	Finalists	1	Rainbow-I-Classic	161,600	103,648	Digital Signature Algorithms
				Rainbow-I-Circumzenithal	60,192	103,648	
				Rainbow-I-Compressed	60,192	64	
			3	Rainbow–III–Classic	882,080	626,048	
				Rainbow–III–Circumzenithal	264,608	626,048	
				Rainbow–III–Compressed	264,608	64	
			5	Rainbow-V-Classic	1,930,600	1,408,736	
				Rainbow-V-Circumzenithal	536,136	1,408,736	
				Rainbow-V-Compressed	536,136	64	
2	GeMSS	Alternates	1 3 5	N/A (No implementation by Open Quantum Safe Project)	352,180	32	Digital Signature Algorithms

Table 6
Runtime analysis of open quantum safe multivariate quadratic equations based cryptographic algorithms (digital signature algorithms).

Algorithm	Keypair/s	Keypair (cycles)	Sign/s	Sign (cycles)	Verify/s	Verify (cycles)
Rainbow-I-Circumzenithal (x86_64)	6.06	412,509,871	530.3	4,714,048	445.7	5,609,619
Rainbow-I-Classic (x86_64)	6.77	369,413,745	506	4,940,751	500.7	4,993,923
Rainbow-I-Compressed (x86_64)	6.02	415,229,320	13.77	181,591,225	445.9	5,607,001
Rainbow–III–Circumzenithal (x86_64)	0.43	1,535,117,154	54.95	45,496,023	49.06	50,968,436
Rainbow–III–Classic (x86_64)	0.49	815,304,452	54.82	45,599,192	50.58	49,426,107
Rainbow–III–Compressed (x86_64)	0.43	1,526,552,696	0.92	2,709,509,453	49.2	50,814,366
Rainbow-V-Circumzenithal (x86_64)	0.15	3,612,623,138	24.87	100,519,687	22.36	111,796,728
Rainbow-V-Classic (x86_64)	0.17	1,503,271,522	23.78	105,150,930	24.24	103,104,686
Rainbow-V-Compressed (x86_64)	0.15	3,653,272,908	0.32	3,503,174,534	21.43	116,636,792

Table 7
Memory Use Analysis (bytes per algorithm) of Open Quantum Safe Multivariate Quadratic Equations Based Cryptographic Algorithms (Digital Signature Algorithms).

Algorithm	Keygen (maxHeap)	Keygen (maxStack)	Sign (maxHeap)	Sign (maxStack)	Verify (maxHeap)	Verify (maxStack)
Rainbow-I-Circumzenithal (x86_64)	168,008	142,440	172,822	912	168,174	324,056
Rainbow-I-Classic (x86_64)	269,416	178,680	274,230	912	274,230	912
Rainbow-I-Compressed (x86_64)	64,424	246,104	64,590	224,504	64,590	324,056
Rainbow–III–Circumzenithal (x86_64)	894,824	946,696	895,088	13,464	895,088	1,765,144
Rainbow–III–Classic (x86_64)	1,512,296	993,416	1,512,832	15,008	1,512,296	384
Rainbow–III–Compressed (x86_64)	268,840	1,572,760	269,104	1,302,104	269,104	1,765,144
Rainbow-V-Circumzenithal (x86_64)	1,949,040	2,208,248	1,949,352	22,024	1949352	3862296
Rainbow-V-Classic (x86_64)	3,343,504	2193512	3343504	384	3343504	384
Rainbow-V-Compressed (x86_64)	540,368	3617000	540,680	2901304	540,680	3862296

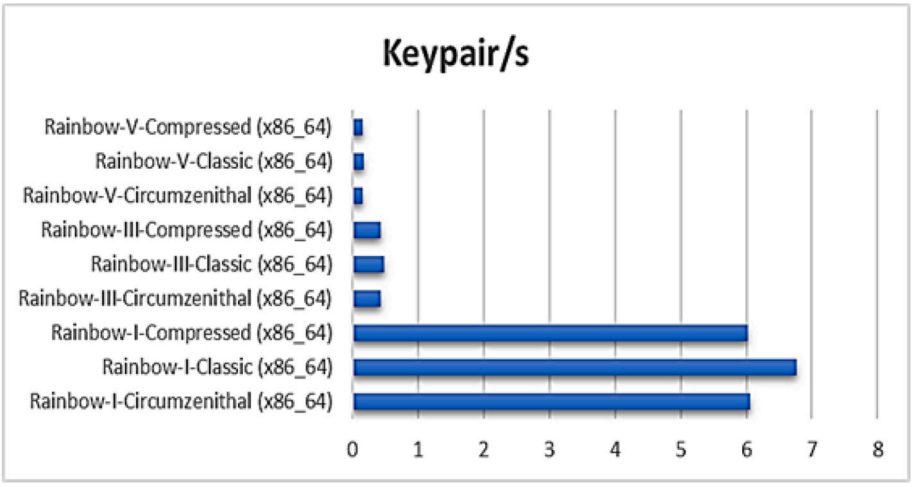

Fig. 12. Keypair operations per second per algorithm (Digital Signature using Multivariate Quadratic Equations Based Algorithms).

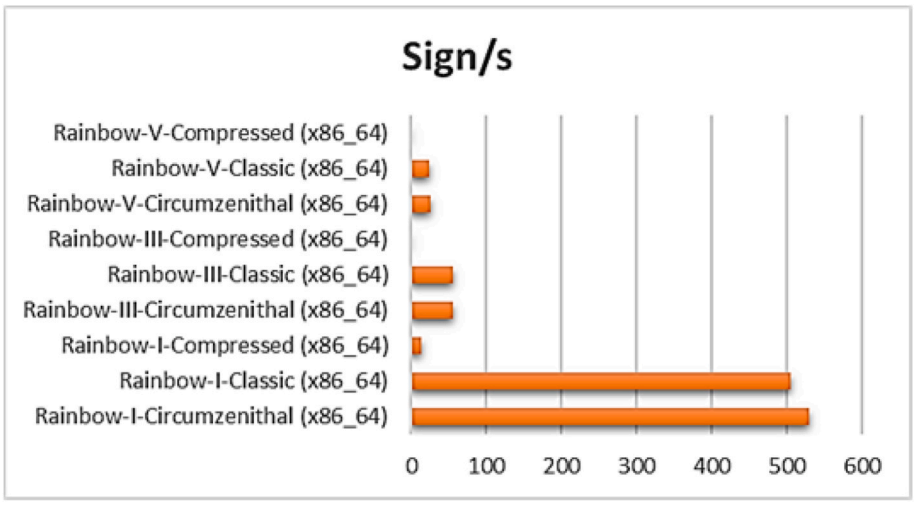

Fig. 13. Sign operations per second per algorithm (Digital Signature using Multivariate Quadratic Equations Based Algorithms).

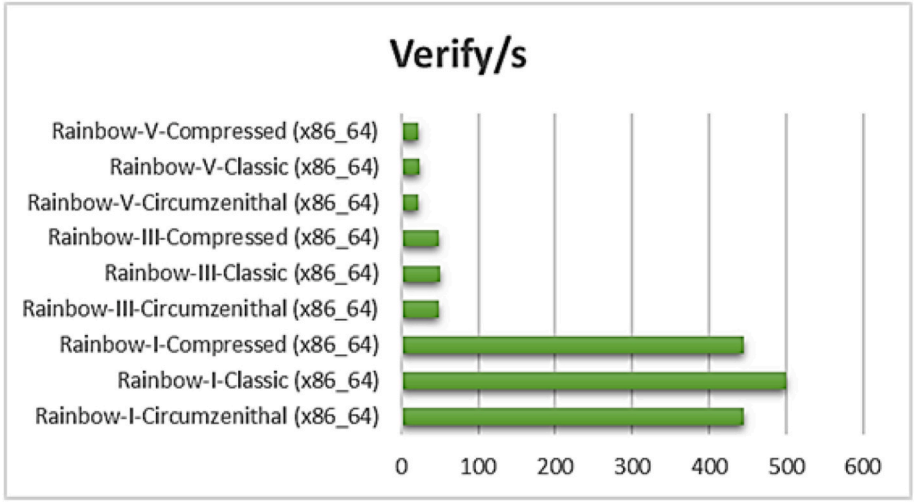

Fig. 14. Verify operations per second per algorithm (Digital Signature using Multivariate Quadratic Equations Based Algorithms).

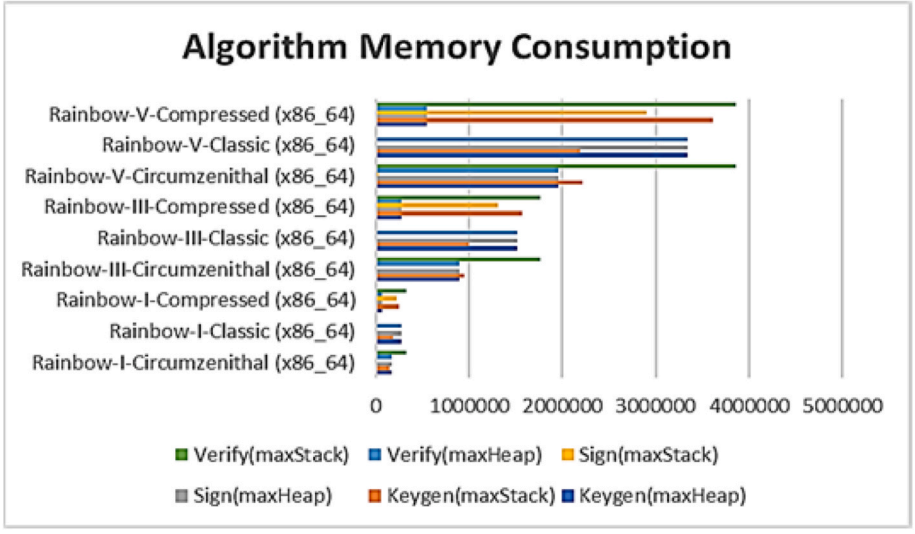

Fig. 15. Memory consumption (bytes per algorithm) (Digital Signature using Multivariate Quadratic Equations Based Algorithms).

Table 8
Hash-based quantum-safe cryptographic algorithms.

Sl. No.	Algorithm Name	Current Status	Open-Source C Library for Quantum-Safe Cryptographic Algorithms (liboqs)				Usage
			NIST Level	Algorithms Name	Public Key Size (bytes)	Private Key Size (bytes)	
1	SPHINCS+	Alternates	1	SPHINCS+-Haraka-128f-robust	32	64	Digital Signature Algorithms
				SPHINCS+-Haraka-128f-simple	32	64	
				SPHINCS+-Haraka-128s-robust	32	64	
				SPHINCS+-Haraka-128s-simple	32	64	
				SPHINCS+-SHA256-128f-robust	32	64	
				SPHINCS+-SHA256-128f-simple	32	64	
				SPHINCS+-SHA256-128s-robust	32	64	
				SPHINCS+-SHA256-128s-simple	32	64	
				SPHINCS+-SHAKE256-128f-robust	32	64	
				SPHINCS+-SHAKE256-128f-simple	32	64	
				SPHINCS+-SHAKE256-128s-robust	32	64	
				SPHINCS+-SHAKE256-128s-simple	32	64	
			3	SPHINCS+-Haraka-192f-robust	48	96	
				SPHINCS+-Haraka-192f-simple	48	96	
				SPHINCS+-Haraka-192s-robust	48	96	
				SPHINCS+-Haraka-192s-simple	48	96	
				SPHINCS+-SHA256-192f-robust	48	96	
				SPHINCS+-SHA256-192f-simple	48	96	
				SPHINCS+-SHA256-192s-robust	48	96	
				SPHINCS+-SHA256-192s-simple	48	96	
				SPHINCS+-SHAKE256-192f-robust	48	96	
				SPHINCS+-SHAKE256-192f-simple	48	96	
				SPHINCS+-SHAKE256-192s-robust	48	96	
				SPHINCS+-SHAKE256-192s-simple	48	96	
			5	SPHINCS+-Haraka-256f-robust	64	128	
				SPHINCS+-Haraka-256f-simple	64	128	
				SPHINCS+-Haraka-256s-robust	64	128	
				SPHINCS+-Haraka-256s-simple	64	128	
				SPHINCS+-SHA256-256f-robust	64	128	
				SPHINCS+-SHA256-256f-simple	64	128	
				SPHINCS+-SHA256-256s-robust	64	128	
				SPHINCS+-SHA256-256s-simple	64	128	
				SPHINCS+-SHAKE256-256f-robust	64	128	
				SPHINCS+-SHAKE256-256f-simple	64	128	
				SPHINCS+-SHAKE256-256s-robust	64	128	
				SPHINCS+-SHAKE256-256s-simple	64	128	

can be rented out for higher computing demand just like cloud computing to serve on a demand basis.

iii) **Third Generation:** Third generation will be true quantum supremacy as the exponential growth and development will bring down the hardware cost. The quantum computer will be affordable and easily accessible to the mass. The third-generation quantum computer will bring the viable solution across a wide variety of non-commercial applications and it will outperform the classical computer and applications.

There are many technological challenges to the development of Quantum Computer with a higher number of qubits. Initially, the breakthrough research in quantum computing was considerably slow and took significant time to realize the working model. However, in the last few years, there is an unprecedented development. Scientists around the world are addressing various challenging issues to develop an affordable quantum computer with higher accuracy [11,48,61]. Gartner in 2018 has published the estimated timeline for physical qubits capacity and quantum computer application in real-life (Fig. 2). If the Gartner timeline is to be believed, society will witness quantum supremacy very soon. Though envisioning the future shows a very good achievement for the computation field, it is a sign of a storm for cyber security.

The development of quantum technology is good news for humanity as it will uplift our life. However, it also raises a serious threat to cybersecurity, requiring a change in how we encrypt our data. As per MIT Technology Review highlights, 20 million-qubit quantum computers could break a 2048-bit number in a mere 8 h [8]. Though the 20 million-qubit quantum computers currently do not exist, we need to be prepared and stay ahead of the threat. We can't wait until those powerful quantum computers start breaking our encryption, it will be too late. It is time to critically analyze the post-quantum threat and prepare accordingly [85,87,90]. We need to identify the research gap and make a strategic plan to breach the gap.

4. Post-quantum encryption problem

Encryption algorithms transform the information in such a way that

Table 9
Runtime analysis of open quantum safe hash based cryptographic algorithms (digital signature algorithms).

Algorithm	Keypair/s	Keypair (cycles)	Sign/s	Sign (cycles)	Verify/s	Verify (cycles)
SPHINCS+-Haraka-128f-robust (x86_64)	3305.33	756,217	130.25	19193087	1841.3	1,357,668
SPHINCS+-Haraka-128f-simple (x86_64)	3021.67	827,241	126.75	19720897	2389.7	1046180
SPHINCS+-Haraka-128s-robust (x86_64)	49.8	50,199,666	6.26	399193532	5109	489,226
SPHINCS+-Haraka-128s-simple (x86_64)	53.95	46336950	6.88	363,156,129	7042	354,926
SPHINCS+-Haraka-192f-robust (x86_64)	2491	1003522	78.92	31672906	1248	2,003,318
SPHINCS+-Haraka-192f-simple (x86_64)	2850	877,072	99.34	25163368	1754.7	1424860
SPHINCS+-Haraka-192s-robust (x86_64)	26.88	92,991,222	2.63	950449292	2971.7	841,268
SPHINCS+-Haraka-192s-simple (x86_64)	30.68	81486323	3.15	794,443,486	3977.3	628,454
SPHINCS+-Haraka-256f-robust (x86_64)	802.67	3114999	36.18	69108207	1203.6	2,076,672
SPHINCS+-Haraka-256f-simple (x86_64)	823.73	3035014	38.42	65055419	1592.1	1569822
SPHINCS+-Haraka-256s-robust (x86_64)	47.11	53,060,953	3.34	748586944	2257.3	1107434
SPHINCS+-Haraka-256s-simple (x86_64)	51.25	48787641	3.93	636,871,172	3061.7	816,421
SPHINCS+-SHA256-128f-robust (x86_64)	962.35	2,597,396	34.83	71789115	216.64	11,537,909
SPHINCS+-SHA256-128f-simple (x86_64)	1833.33	1363722	65.69	38055893	410.73	6,087,008
SPHINCS+-SHA256-128s-robust (x86_64)	15.42	162114391	1.94	1289794883	574.14	4354738
SPHINCS+-SHA256-128s-simple (x86_64)	27.71	90,208,011	3.64	687556446	1228.3	2035202
SPHINCS+-SHA256-192f-robust (x86_64)	676.11	3697467	22.13	112,975,498	144.19	17338755
SPHINCS+-SHA256-192f-simple (x86_64)	1200.33	2082895	38.49	64952624	250.42	9983430
SPHINCS+-SHA256-192s-robust (x86_64)	10.41	240,149,708	1.08	2317742108	399.4	6258715
SPHINCS+-SHA256-192s-simple (x86_64)	18.74	133365900	1.88	1,331,847,740	730.33	3423401
SPHINCS+-SHA256-256f-robust (x86_64)	140.24	17824497	6.67	374615844	108.67	23007612
SPHINCS+-SHA256-256f-simple (x86_64)	450.85	5,544,896	20.81	120145814	272.91	9159588
SPHINCS+-SHA256-256s-robust (x86_64)	8.73	286251953	0.75	3,329,041,447	207.26	12062042
SPHINCS+-SHA256-256s-simple (x86_64)	30.34	82391795	2.4	1043537732	530.33	4,714,332
SPHINCS+-SHAKE256-128f-robust (x86_64)	462.36	5,406,859	18.56	134699425	154.69	16161667
SPHINCS+-SHAKE256-128f-simple (x86_64)	844	2962241	32.16	77,738,565	301.8	8283439
SPHINCS+-SHAKE256-128s-robust (x86_64)	7.81	320,032,253	1.02	2460642112	543.49	4,599,201
SPHINCS+-SHAKE256-128s-simple (x86_64)	13.66	183,072,418	1.71	1463603275	915.67	2730203
SPHINCS+-SHAKE256-192f-robust (x86_64)	331.89	7532419	11.55	216,435,158	119.17	20977226
SPHINCS+-SHAKE256-192f-simple (x86_64)	580.14	4309068	20.11	124327976	229.26	10,905,347
SPHINCS+-SHAKE256-192s-robust (x86_64)	5.2	480506759	0.58	46486796	319.23	7831342
SPHINCS+-SHAKE256-192s-simple (x86_64)	9.35	267,514,170	1	2498600938	655.33	3,814,860
SPHINCS+-SHAKE256-256f-robust (x86_64)	123.63	20220286	5.92	422,251,215	108.78	22981847
SPHINCS+-SHAKE256-256f-simple (x86_64)	220.11	11,357,895	10.37	241136741	209.39	11,937,656
SPHINCS+-SHAKE256-256s-robust (x86_64)	7.5	333331669	0.66	3783521422	227.85	10973738
SPHINCS+-SHAKE256-256s-simple (x86_64)	13.28	188,315,722	1.08	2311104060	427.19	5852661

it is difficult to interpret it unless and until it is decrypted using the decryption key [27,33,86]. Most of the encryption algorithms are based on complex mathematical function (Fig. 3) which is easy to compute in one direction but very difficult to compute in the reverse direction [23, 28]. These algorithms are known as one-way functions. RSA is one of the most commonly used encryption algorithms which makes use of the one-way function.

The core strength of the RSA algorithm is based on prime factorization as a method of one-way encryption. The sender uses an encryption key which is generated using the multiplication of two randomly generated large prime numbers. The receiver can decrypt the message, only if knows the exact prime factor. The problem is easy to solve in one direction but difficult to compute in another direction.

Suppose we want to factor an integer N with d decimal digits. The brute force algorithm goes through all primes p up to $\sqrt{N}$ and checks whether p divides N. The worst-case time complexity of the brute force algorithm is approximately $\sqrt{N}$, which is exponential in the number of digits d.

The quadratic sieve algorithm, construct integers a, b such that $a^2 - b^2$ is a multiple of N. Once such a combination of a, b is found, one checks whether have $a \pm b$ common factors with N . The quadratic sieve is very efficient it has asymptotic runtime complexity in $\sqrt{d}$. The generic number field sieve, the most efficient classical factoring technique, achieves an asymptotic runtime exponential in $b^{1/3}$.

The traditional factoring methods' usefulness is limited to numbers with a few hundred digits due to exponential runtime scaling. Shor's factoring algorithm, on the other hand, has a runtime polynomial in d (number of digits). The performance of classical vs. quantum algorithms for prime number factorization is shown in Fig. 4 [53].

4.1. The effect of an algorithm

In 1994 Peter Shor developed a polynomial-time quantum computer algorithm for integer factorization. It was a major breakthrough to prove the Quantum Computer's supremacy over a classical computer [37,46]. Shor's algorithm will greatly reduce the prime factorization time. All the encryption algorithm which is based on the assumption that extracting the private key is computationally infeasible with the existing computing resource does not stand.

Successful implementation of Shor's algorithm could easily break the most widely used cryptographic algorithm such as RSA, Elliptic Curve Cryptography (ECC), and Diffie-Hellman. These algorithms play a vital role in data protection and privacy. Any possibility to break these algorithms is a serious threat and could be devastating [15,21,73]. It will be a direct threat to the information and communication security of individuals, government, and business organizations.

In 1996 Lov Grover developed an algorithm using quantum computers to search the unsorted database. Grover's search algorithm can find an element in the unsorted database of N elements in $\sqrt{N}$ searches. The algorithm can search and find the key in very less iterations. For example, Grover's search algorithm needs only 185 searches to find the key for the DES algorithm which relies on a 56-bit key [49]. Though Grover's search algorithm is not so fast as Shor's algorithm it is a potential threat to the symmetric cryptography algorithm. If a Quantum Computer of sufficient size and fault tolerance is built, then Shor's algorithm will break today's public-key cryptography. Variants of the algorithm are applied to both the discrete logarithm problem (which breaks elliptic curve cryptography and finite field Diffie-Hellman) and to factoring large integers (which breaks RSA) (Fig. 5).

Table 10
Memory Use Analysis (bytes per algorithm) of Open Quantum Safe Hash-Based Cryptographic Algorithms (Digital Signature Algorithms).

Algorithm	Keygen (maxHeap)	Keygen (maxStack)	Sign (maxHeap)	Sign (maxStack)	Verify (maxHeap)	Verify (maxStack)
SPHINCS+-Haraka-128f-robust (x86_64)	8912	3712	26,100	19,472	26,100	2304
SPHINCS+-Haraka-128f-simple (x86_64)	8912	3712	26,100	19,472	26,100	2304
SPHINCS+-Haraka-128s-robust (x86_64)	8912	2368	16,868	10,240	16,868	2304
SPHINCS+-Haraka-128s-simple (x86_64)	8912	2368	16,868	10,240	16,868	2304
SPHINCS+-Haraka-192f-robust (x86_64)	8960	2432	44,724	38,128	44,724	1960
SPHINCS+-Haraka-192f-simple (x86_64)	8960	2432	44,724	38,128	44,724	1960
SPHINCS+-Haraka-192s-robust (x86_64)	8960	2432	25,284	18,672	25,284	2304
SPHINCS+-Haraka-192s-simple (x86_64)	8960	2432	25,284	18,672	25,284	2304
SPHINCS+-Haraka-256f-robust (x86_64)	9008	2496	58,964	52,352	58,964	1960
SPHINCS+-Haraka-256f-simple (x86_64)	9008	2496	58,964	52,352	58,964	1960
SPHINCS+-Haraka-256s-robust (x86_64)	4360	7400	34,252	38,552	38,900	1960
SPHINCS+-Haraka-256s-simple (x86_64)	4360	7400	34,252	37,800	38,900	1960
SPHINCS+-SHA256-128f-robust (x86_64)	4432	9416	21,620	11,752	26,100	2304
SPHINCS+-SHA256-128f-simple (x86_64)	4432	7560	21,620	9928	26,100	2304
SPHINCS+-SHA256-128s-robust (x86_64)	4432	9640	12,388	12,840	16,868	2304
SPHINCS+-SHA256-128s-simple (x86_64)	4432	7784	12,388	11,016	16,868	2304
SPHINCS+-SHA256-192f-robust (x86_64)	4480	10,536	40,244	14,120	40,412	6816
SPHINCS+-SHA256-192f-simple (x86_64)	4480	8488	40,244	12,072	44,724	1960
SPHINCS+-SHA256-192s-robust (x86_64)	4480	10,856	20,804	15,944	25,284	2304
SPHINCS+-SHA256-192s-simple (x86_64)	4480	8808	20,804	13,896	25,284	2304
SPHINCS+-SHA256-256f-robust (x86_64)	4528	11,944	54,484	16,360	54,652	7840
SPHINCS+-SHA256-256f-simple (x86_64)	4528	9704	54,484	14,152	54,652	7840
SPHINCS+-SHA256-256s-robust (x86_64)	4528	12,200	34,420	18,376	34,588	7840
SPHINCS+-SHA256-256s-simple (x86_64)	4528	9992	34,420	16,168	34,588	6720
SPHINCS+-SHAKE256-128f-robust (x86_64)	8912	2432	26,100	18,192	26,100	2304
SPHINCS+-SHAKE256-128f-simple (x86_64)	8912	2432	26,100	18,192	26,100	2304
SPHINCS+-SHAKE256-128s-robust (x86_64)	8912	2560	13,052	13,128	16,868	2304
SPHINCS+-SHAKE256-128s-simple (x86_64)	8912	2560	13,052	13,064	16,868	2304
SPHINCS+-SHAKE256-192f-robust (x86_64)	8960	2432	40,908	41,608	44,724	1960
SPHINCS+-SHAKE256-192f-simple (x86_64)	8960	2432	40,908	41,608	44,724	1960
SPHINCS+-SHAKE256-192s-robust (x86_64)	8960	2432	21,468	22,888	25,284	2304
SPHINCS+-SHAKE256-192s-simple (x86_64)	8960	2432	21,468	22,888	25,284	2304
SPHINCS+-SHAKE256-256f-robust (x86_64)	9008	2496	55,148	57,000	58,964	1960
SPHINCS+-SHAKE256-256f-simple (x86_64)	9008	2496	55,148	57,000	58,964	1960
SPHINCS+-SHAKE256-256s-robust (x86_64)	5192	6464	35,084	37,800	38,900	1960
SPHINCS+-SHAKE256-256s-simple (x86_64)	5192	6464	35,084	37,800	38,900	1960

4.2. Sense of urgency

As described by Dr. Michele Mosca [47], the urgency for any organization to transit to the quantum-resistant or quantum-safe cryptography depends on the following parameters:

- **Shelf-Life Time (X Years):** Number of years you need your cryptographic key to remain secure and your data to be protected.
- **Migration Time (Y Years):** Number of years required to develop, deploy and migrate to the Quantum-Safe solution.
- **Threat Timeline (Z Years):** The number of years before large-scale quantum computers will be built which can break the current cryptography algorithms.

If the Threat Timeline is shorter than the sum of Shelf-Life Time and Migration Time (Fig. 6) i.e. 'X + Y > Z' then it is a matter of serious concern as the organizations will not be able to protect their assets for the required years against quantum attack [5,62,65].

Assessing the exact Threat Timeline (i.e., value for 'Z') is a challenging task as there are so many obstacles in building the quantum computer with a required number of qubits and efficiency. However, the current trend in research and development shows that the days are not far away when we will have quantum computers with the required computation power [9,19,50]. At some point, we can also expect Moore's Law to support the scaling up of the development of quantum Computer technology as it happened with conventional computing technologies.

The threat is not only challenging for future perspective but it is significant today itself [24,88]. It is quite possible that many hackers, state-sponsored or nation-states themselves might be intercepting and harvesting the encrypted messages with envisioning that they will be able to decrypt these messages when the quantum computing resources will be available in the future. If today's private messages get disclosed even in the future, it will have a serious adverse effect on corporates, government organizations, the military, and the diplomatic relations of countries. So, the threat is not in the future as shown by Mosca Theorem but has already begun from the very first day Shor's algorithm was introduced in 1994. Hence, any communication done since 1994 using an existing cryptography algorithm that is not quantum-safe is vulnerable and can be exposed in the future [7,63].

4.3. Quantum computing impact on existing cryptography algorithm

A practical quantum computer with the required number of qubits (millions of qubits) will be able to break all the modern public-key cryptographic systems. National Institute of Standards and Technology (NIST) 2016 report on post-quantum cryptography shown in Table 1, highlights the impact of quantum computing on common cryptographic algorithms.

NIST is rigorously working to analyze, test, and validate post-quantum algorithms and is expected to release a draft standard by 2023. The potential algorithms are evaluated on various factors such as robustness against cryptanalysis attacks, algorithm efficiency, interoperability, implementation feasibility, etc.

5. Global initiatives and standardization

The importance of the Quantum-Safe algorithm is well understood by government organizations and IT giants around the world. The threat is real and that day is fast approaching. Hence many countries and

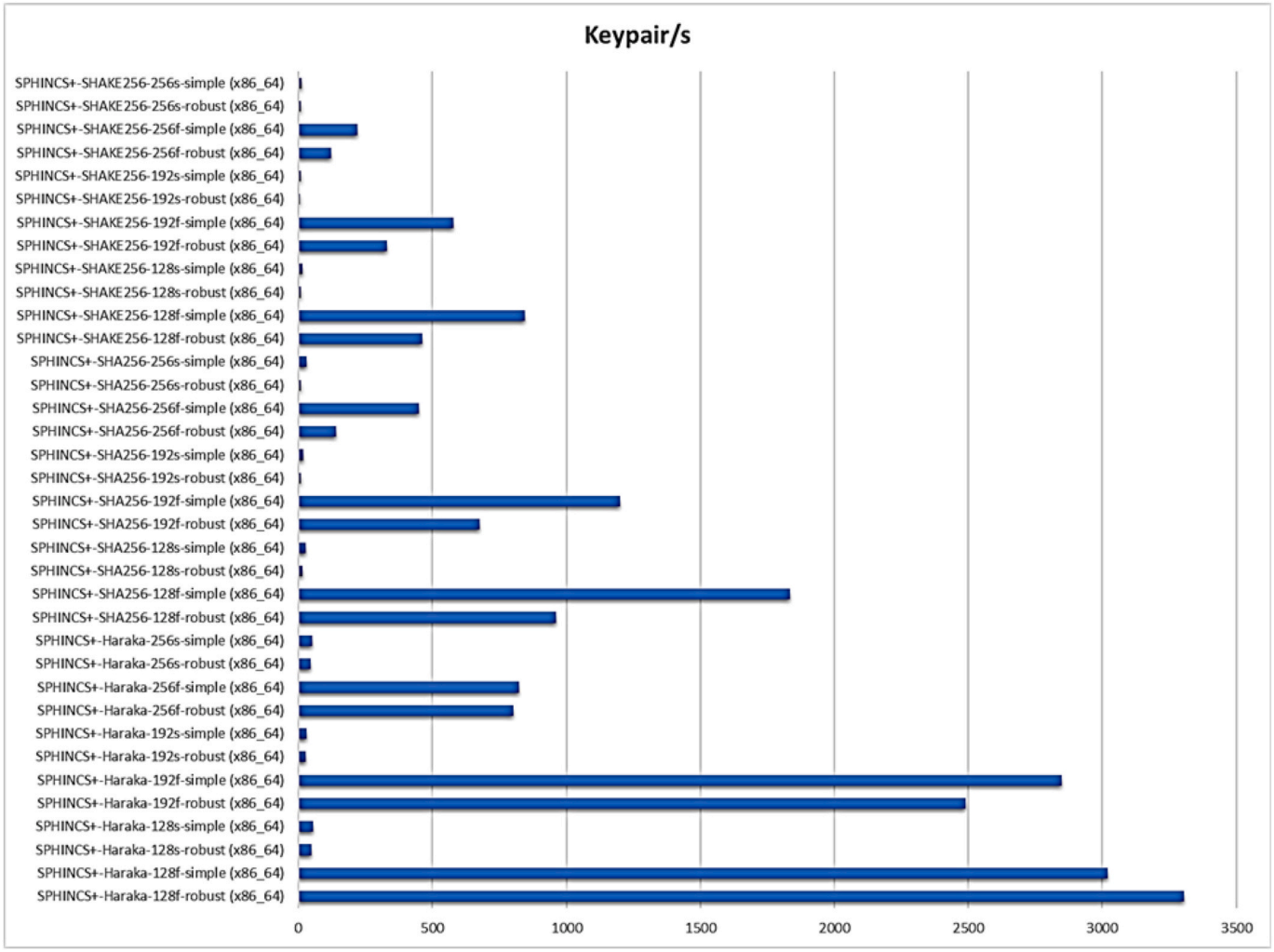

Fig. 16. Keypair operations per second per algorithm (Digital Signature using Hash-Based Algorithms).

standard organizations have taken an initiative in the design, development, testing, and migration strategy for quantum-safe algorithms [56]. The working group, forum, and standard organizations are publishing framework, policy documents, whitepapers, technical details, standardization documents, and cost-effective migration strategies to post-quantum cryptography. Some of the organizations leading the standardization process are:

i) **National Institute of Standards and Technology (NIST) PQC Standardization:** The major initiative has been taken by the National Institute of Standards and Technology (NIST) to solicit, evaluate and standardize the quantum-resistant public-key cryptographic algorithms. The NIST has begun the process in 2016 to determine the criteria and requirements for quantum-safe algorithms and announced the call for proposal [71]. The first round of submissions was announced in 2017 in which 82 submissions were received out of which 69 were announced as 1st round candidates. Analysis for the 1st round of candidates was done in 2018 and 1st NIST QPC standardization conference was held. The 2nd NIST PQC standardization conference was held in the year 2019 and 26 candidates were announced for the 2nd round. The NIST announced 7 finalists and 8 alternate candidates during the 3rd round in the year 2020 [2]. NIST completed the 3rd round process in July 2022. Four algorithms have been selected for standardization and additional four algorithms have been considered for the 4th round of evaluation [3,6]. The plausible timeline for NIST PQC standards development and adoption is shown in Fig. 7 [77].

ii) **The European Telecommunications Standards Institute (ETSI):** The European Telecommunications Standards Institute (ETSI) has a Cyber Quantum-Safe Cryptography (QSC) group, actively working on post-quantum-safe algorithms. The QSC working groups aim to recommend a quantum-safe cryptographic algorithm and its implementation. The focus is on architectural consideration for specific applications, implementation capabilities, performance, etc. The group has published cost-effective migration strategies to post-quantum cryptography and some of the challenges and recommendations. ETSI is also organizing workshops on post-quantum cryptography since 2013.

iii) **The Internet Engineering Task Force (IETF):** The Internet Engineering Task Force (IETF) is focused on promoting voluntary internet standards. IETF has Crypto Forum Research Group (CFRG). The CFRG forum bridge the gap between industry and research organizations, bringing new cryptographic techniques to the Internet community and promoting an understanding of the use and applicability of these mechanisms via Informational RFCs. The working group is developing Request for Comment (RFC) protocols compatible with post-quantum cryptography.

iv) **The American National Standards Institute (ANSI):** The Accredited Standards Committee X9F which is the subcommittee of the American National Standards Institute (ANSI) has made some recommendations on lattice-based quantum-safe

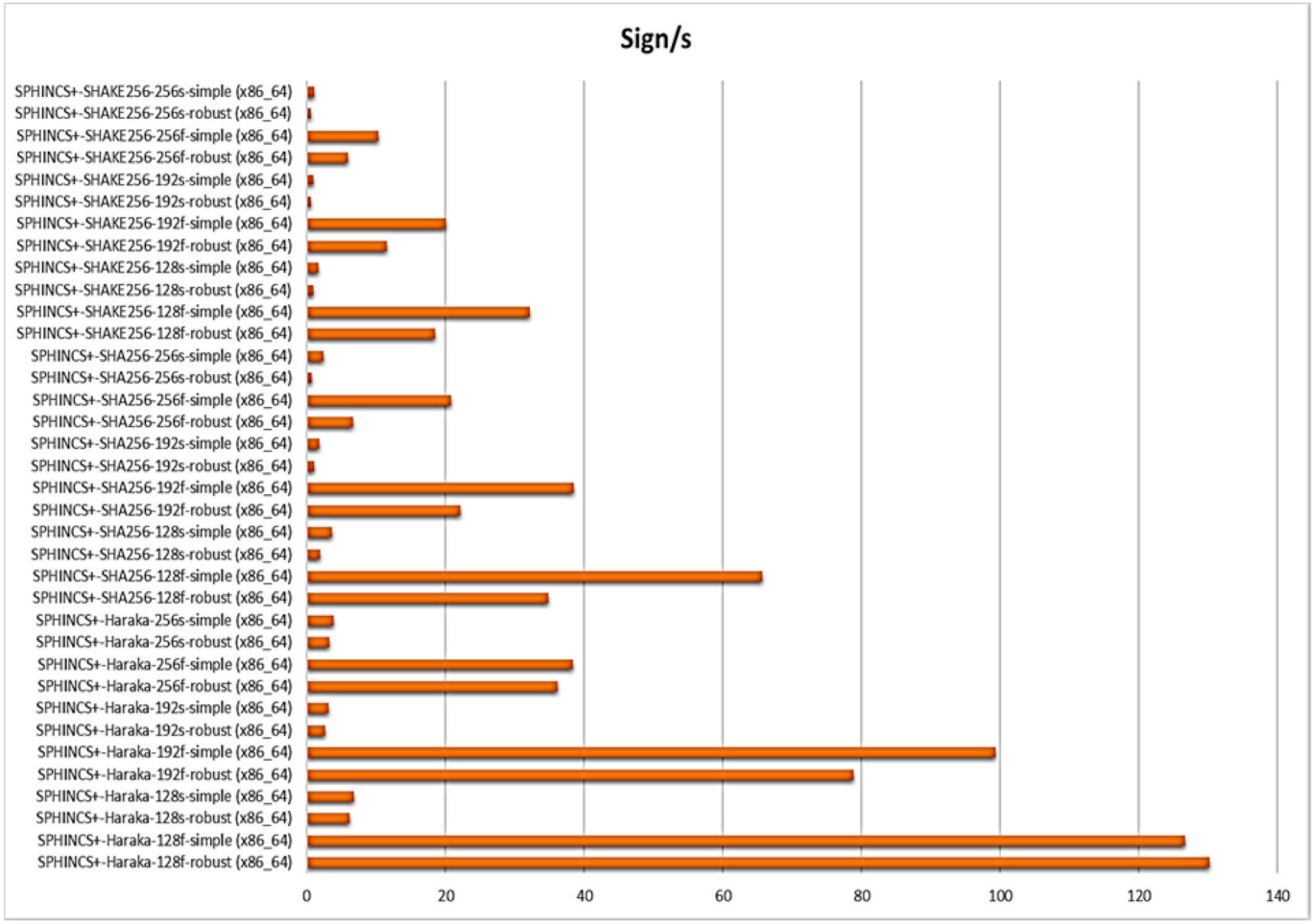

Fig. 17. Sign operations per second per algorithm (Digital Signature using Hash-Based Algorithms).

algorithms as far back as 2010. The X9 committee is working actively on methods to minimize or eliminate avenues of attack by quantum computers on the financial services industry. The working group's objective is to determine what preventive measures the financial service industry should be taking to protect against attacks from a large-scale quantum computer and publish periodical reports and whitepapers.

v) **Federal Office for Information Security (BSI):** In order to be prepared for future risk, the German Federal Office for Information Security, BSI has also released technical guidelines that recommend algorithms, and key lengths and make specific recommendations for Merkle-based signature schemes and XMSS. The focus is the development of quantum-safe public-key cryptography algorithms. In 2020 the BSI also developed the initial recommendations for the migration to post-quantum cryptography.

6. Potential solution for post quantum encryption challenges

Scientists and research organizations are working very hard to develop and standardize quantum-safe cryptographic algorithms. However, not much information is available in the public domain except for the work done by NIST. The NIST process to solicit, evaluate, and standardize the quantum-resistant cryptographic algorithms is very meticulous, and detailed information is published. Hence, decided to do a comparative analysis of 7 finalists and 8 alternate quantum-resistant algorithms announced by the NIST during the 3rd round in the year 2020. The objective of the analysis is to assess the implementation feasibility of the prospective algorithms based on their CPU cycle and memory utilization.

The performance analysis of the algorithms is done using the Open Quantum Safe (OQS) project. It is a project, developing and prototyping quantum-resistant cryptography algorithms [4]. OQS has developed "liboqs" which is an open-source library for quantum-resistant cryptographic algorithms. OQS is mainly focused on the NIST Post-Quantum Cryptography standardization for Key Encryption Mechanisms (KEM) and Signature Schemes. The OQS also provides benchmarking data for various quantum-resistant algorithms which are used in this paper for the comparative analysis of the algorithm's runtime behavior and memory consumption. The runtime behavior and memory consumption data of the algorithms are collected based on the execution of the algorithms on Amazon Web Service (AWS) with CPU Model Intel(R) Xeon (R) Platinum 8259CL CPU @ 2.50 GHz.

6.1. Code-based cryptography

The code-based cryptography algorithm (e.g., McEliece) relies on the hardness of decoding in a linear error-correcting code possibly chosen with some particular structure or in a specific family (for instance, quasi-cyclic codes, or Goppa codes). Classic McEliece, a code-based post-quantum public-key cryptosystem (PKC) contender for NIST's world-wide standardization, was suggested by Daniel J. Bernstein in 2017 [17,18,79]. In McEliece's idea, a public key is the product of the Goppa code and the linear transformation. To encrypt the message, the sender needs to add a specific amount of random noise [1,12–14]. The noise can be removed only using the Goppa code [66,70]. It is a computationally

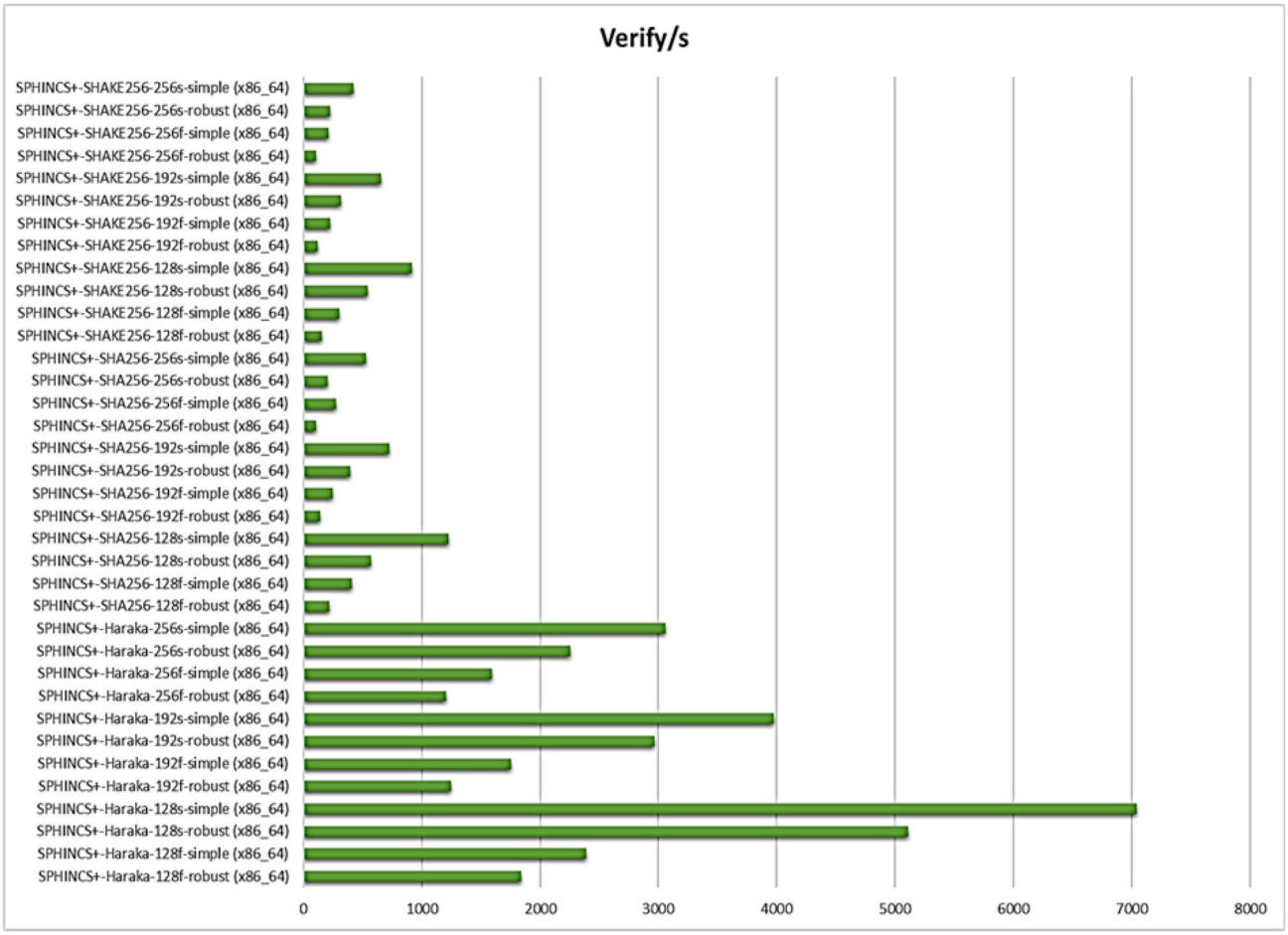

Fig. 18. Verify operations per second per algorithm (Digital Signature using Hash-Based Algorithms).

challenging problem for the attacker to recover the message without knowing how to factor in the public key. There are various code-based cryptography algorithms available, among which some potential quantum-safe algorithms are:

i) **Classic McEliece** was introduced by Robert McEliece in 1978. It is a code-based algorithm proposed for the Key Encapsulation Mechanism (KEM). There are very few changes in the algorithm since it was introduced. The changes in the algorithm's security parameters are done to increase the computational speeds. Classic McEliece parameters can be set to match all the five NIST security levels. The computation time of the Classic McEliece algorithm is extremely fast but the algorithms require a very large public key size.
ii) **BIKE** (Bit Flipping Key Encapsulation) is a code-based key encapsulation mechanism. It is based on quasi-cyclic moderate density parity-check (QC-MDPC) codes that can be decoded using bit flipping decoding techniques [68]. BIKE can satisfy the NIST security level 1 and 3.
iii) **HQC** (Hamming Quasi-Cyclic) is a code-based public-key algorithm designed to provide security against classical and quantum computers. The HQC algorithm is designed to achieve NIST 1, 3, and 5 security levels [51].

Classic McEliece is a round 3 finalist whereas BIKE, and HQC are declared as alternate candidates by NIST (Table 2). Runtime behavior and memory consumptions of Classic MeEliece, BIKE, and HQC are shown in Table 3, Table 4, and graphically represented in Fig. 8, Fig. 9, Fig. 10, and Fig. 11.

6.2. *Multivariate Quadratic Equations based cryptography*

A multivariate quadratic equation-based public key cryptosystem is having a set of quadratic polynomials over a finite field. In some cases, those polynomials could be defined over both grounds and as an extension field. The security of a multivariate public-key cryptosystem is based on the fact that solving the multivariate polynomial equation is proven to be NP-Hard or NP-Complete. Cryptographic algorithms based on Multivariate Quadratic Equations which are potentially quantum-safe are as follows:

i) **Rainbow** algorithms belong to the family of multivariate public-key cryptosystems. It is designed by Jintai Ding and Dieter Schmidt in 2004. The algorithm is based on the Oil-Vinegar signature scheme invented by Jacques Patarin. The algorithm is based on the fact that solving a set of random multivariate quadratic systems is an NP-hard problem [20,72]. The algorithm can be used for digital signature and its parameters can be set to achieve NIST security levels 1, 3, and 5.
ii) **GeMSS** (A Great Multivariate Short Signature) is a multivariate-based signature scheme producing small signatures. GeMSS is based on the Hidden Field Equations cryptosystem (HFE) using the minus and vinegar modifiers i.e., HFEv- [25,44]. The algorithm has a medium/large public key and it has a very fast verification process.

Rainbow is announced as a finalist for round 3 and GeMSS is announced as an alternate (Table 5). The runtime behavior and memory consumptions of the Rainbow algorithm are shown in Table 6, Table 7, and graphically represented in Fig. 12, Fig. 13, Fig. 14, and Fig. 15. No implementation is available for GeMSS by Open Quantum Safe Project.

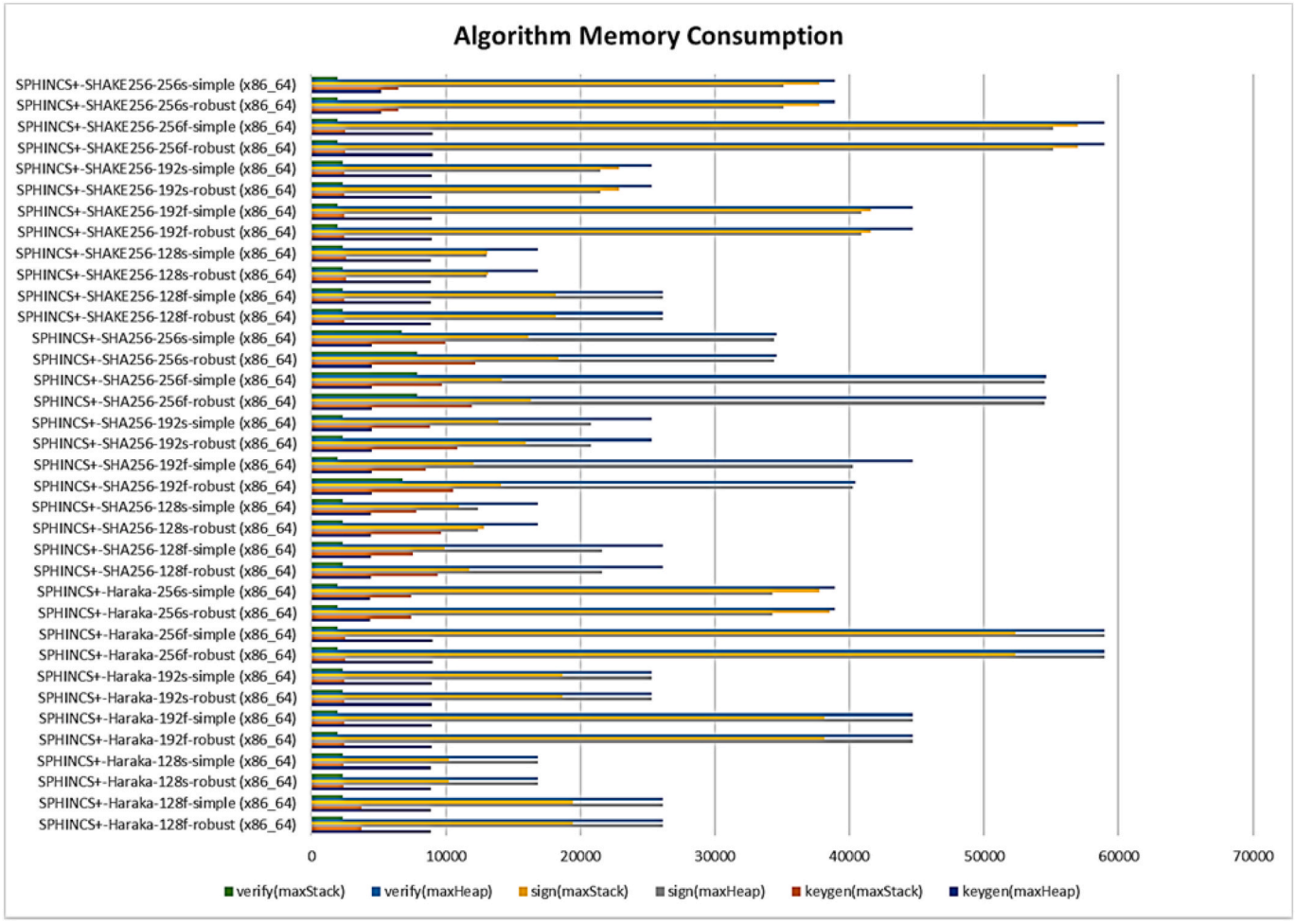

Fig. 19. Memory consumption (bytes per algorithm) (Digital Signature using Hash-Based Algorithms).

Table 11
Isogeny-based quantum-safe cryptographic algorithms.

Sl. No.	Algorithm Name	Current Status	Open-Source C Library for Quantum-Safe Cryptographic Algorithms (liboqs)				Usage
			NIST Level	Algorithms Name	Public Key Size (bytes)	Private Key Size (bytes)	
1	SIKE	Alternates	1	SIDH-p434	330	28	Key Encapsulation Mechanisms
				SIDH-p434-compressed	197	28	
				SIKE-p434	330	374	
				SIKE-p434-compressed	197	350	
			2	SIDH-p503	378	32	
				SIDH-p503-compressed	225	32	
				SIKE-p503	378	434	
				SIKE-p503-compressed	225	407	
			3	SIDH-p610	462	39	
				SIDH-p610-compressed	274	39	
				SIKE-p610	462	524	
				SIKE-p610-compressed	274	491	
			5	SIDH-p75	564	48	
				SIDH-p751-compressed	335	48	
				SIKE-p751	564	644	
				SIKE-p751-compressed	335	602	

However, the detailed performance analysis for GeMSS is available on GeMSS official website [44].

6.3. Hash-based cryptography

Cryptographic hash functions are non-reversible function that takes a string of any length as input and produces a fixed-length output. Hash-based cryptography is popularly used for digital signatures such as Merkle signature. It combines a one-time signature with the Merkle tree. The hash-based signature (HBS) is a collection of many one-time signature (OTS) schemes. HBS uses a tree data structure to combine many OTS in an efficient manner. To sign a message, an HBS chooses a single OTS from its collection and uses it to sign the message [29]. The critical issue is that the HBS must never choose the same OTS twice, or else security is compromised. Hash-based quantum-safe cryptographic algorithms are:

Table 12
Runtime analysis of open quantum safe isogeny based cryptographic algorithms (Key encapsulation mechanisms).

Algorithm	Keygen/s	Keygen (cycles)	Encaps/s	Encaps (cycles)	Decaps/s	Decaps (cycles)
SIDH-p434 (x86_64)	109.63	22805106	53.61	46626958	135.2	18488925
SIDH-p434-compressed (x86_64)	57.09	43,797,293	38.69	64616727	119.7	20885902
SIDH-p503 (x86_64)	318.89	7840371	155.4	16,085,706	391.2	6389639
SIDH-p503-compressed (x86_64)	166.44	15020665	111.7	22385569	330.9	7,555,453
SIDH-p610 (x86_64)	30.91	80887652	15.93	156909224	37.1	67391175
SIDH-p610-compressed (x86_64)	18.88	132,403,975	13.28	188156935	36.07	69302679
SIDH-p751 (x86_64)	104.79	23,858,838	50.17	49,829,353	127.4	19629441
SIDH-p751-compressed (x86_64)	55.93	44696733	36.57	68372097	114.3	21,873,987
SIKE-p434 (x86_64)	98.8	25300016	60.57	41,275,416	56.52	44237605
SIKE-p434-compressed (x86_64)	57.71	43322569	36.58	68330870	52.79	47,358,432
SIKE-p503 (x86_64)	271.82	9197872	175	14283364	163	15,339,004
SIKE-p503-compressed (x86_64)	165.89	15,070,206	112.1	22309024	154.3	16200750
SIKE-p610 (x86_64)	30.84	81071205	16.84	148,430,862	16.75	149271935
SIKE-p610-compressed (x86_64)	18.99	131620069	13.28	188252832	16.99	147,137,618
SIKE-p751 (x86_64)	92.91	26906560	57.49	43,480,834	53.45	46777582
SIKE-p751-compressed (x86_64)	55.8	44,799,410	36.53	68,434,047	47.73	52,377,374

Table 13
Memory Use Analysis (bytes per algorithm) of Open Quantum Safe Isogeny Based Cryptographic Algorithms (Key Encapsulation Mechanisms).

Algorithm	Keygen (maxHeap)	Keygen (maxStack)	Encaps (maxHeap)	Encaps (maxStack)	Decaps (maxHeap)	Decaps (maxStack)
SIDH-p434 (x86_64)	4534	9928	5002	10,376	5084	9576
SIDH-p434-compressed (x86_64)	4401	78,248	4736	57,624	4818	9576
SIDH-p503 (x86_64)	9234	2192	9738	2320	9864	2320
SIDH-p503-compressed (x86_64)	4433	92,456	4816	63,840	9558	2304
SIDH-p610 (x86_64)	4677	14,520	5332	15,112	5447	14,008
SIDH-p610-compressed (x86_64)	4489	146,840	4956	104,072	5071	15,352
SIDH-p751 (x86_64)	4788	11,416	5588	12,216	5728	10,440
SIDH-p751-compressed (x86_64)	4559	196,440	5130	123,944	5270	10,760
SIKE-p434 (x86_64)	4880	10,152	5242	10,504	5258	11,192
SIKE-p434-compressed (x86_64)	4723	78,248	4975	58,072	4991	15,912
SIKE-p503 (x86_64)	9636	2192	10,062	2832	5438	7768
SIKE-p503-compressed (x86_64)	4808	92,456	5112	64,368	5136	13,784
SIKE-p610 (x86_64)	5162	14,856	5672	15,320	5696	16,248
SIKE-p610-compressed (x86_64)	4941	146,840	5301	104,712	5325	25,144
SIKE-p751 (x86_64)	5384	12,200	6012	12,376	6044	12,792
SIKE-p751-compressed (x86_64)	5113	196,456	5555	124,696	5587	22,440

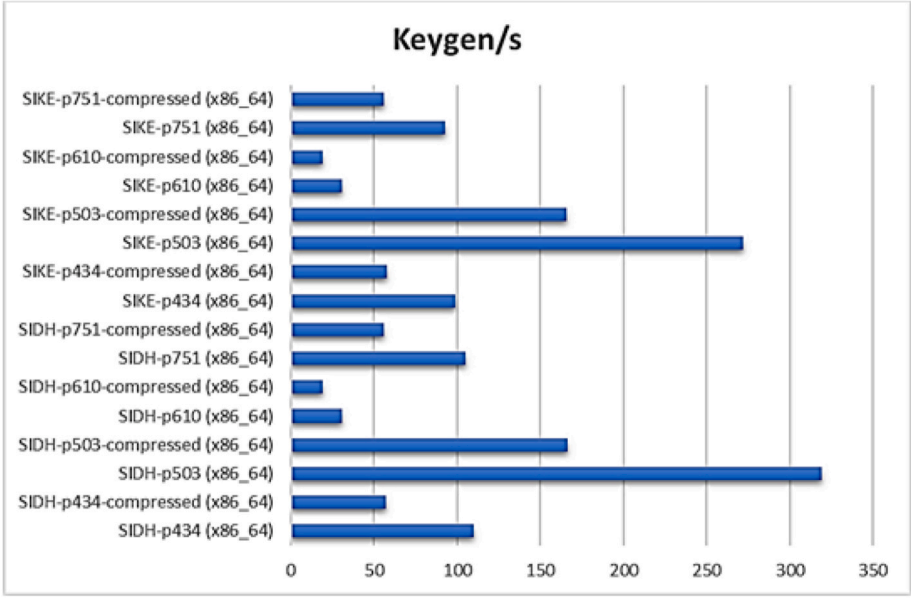

Fig. 20. Keygen operations per second per algorithm (Key Encapsulation Mechanisms using Isogeny Based Algorithms).

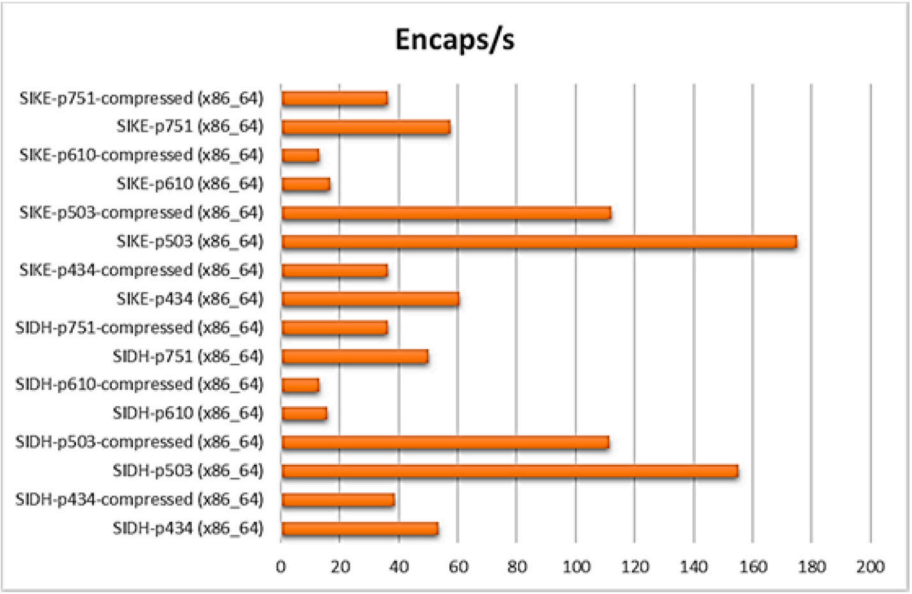

Fig. 21. Encaps operations per second per algorithm (Key Encapsulation Mechanisms using Isogeny Based Algorithms).

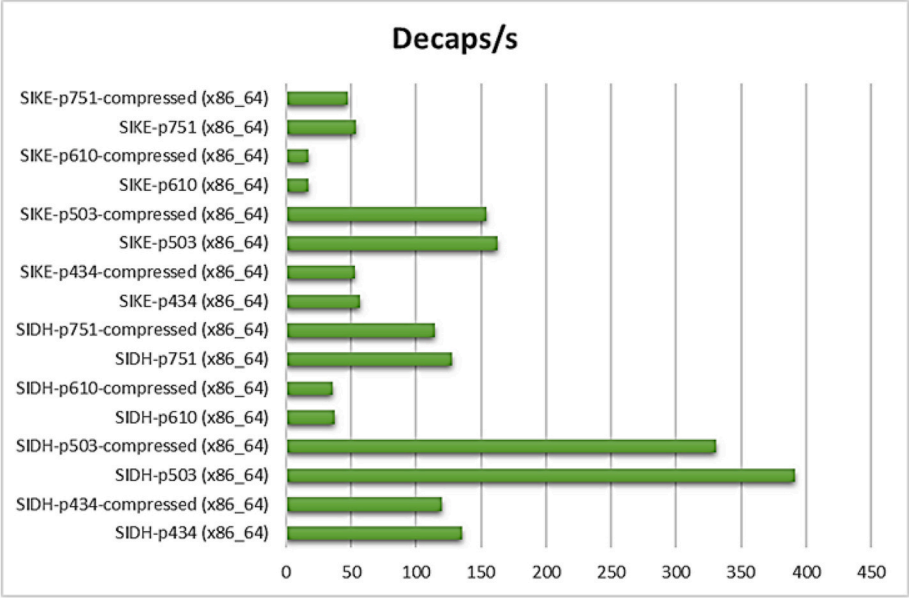

Fig. 22. Decaps operations per second per algorithm (Key Encapsulation Mechanisms using Isogeny Based Algorithms).

i) **SPHINCS+** is a stateless hash-based signature scheme. It is an improvised version of SPHINCS, especially aimed to reduce the signature size [80]. The NIST submission of SPHINCS+ proposes three different signature schemes:
SPHINCS+-SHAKE256
SPHINCS+-SHA-256
SPHINCS+-Haraka

SPHINCS+ is announced as an alternate by NIST in the 3rd round of algorithm selection (Table 8). The algorithm can be used for digital signatures and can satisfy the NIST 1, 3, and 5 security levels. Runtime behavior and memory consumptions of SPHINCA+ are shown in Table 9, Table 10, and graphically represented in Fig. 16, Fig. 17, Fig. 18, and Fig. 19.

6.4. Isogeny based cryptography

It is a relatively new technique that has begun in the year 2000. Isogeny-based cryptography uses maps between elliptic curves to build public-key cryptosystems. The security of Isogeny cryptography is based on so-called supersingular isogeny problems or in other words finding the isogeny mapping between two supersingular elliptic curves with the same number of points. The Isogeny-based protocols require a very small key compared to any other post-quantum cryptography candidates. SIKE is the one potential quantum-safe algorithm belonging to Isogeny cryptography family [30,40,82].

i) **SIKE** (Supersingular Isogeny Key Encapsulation) is an isogeny-based key encapsulation algorithm. The algorithm implementation is based on pseudo-random walks in supersingular isogeny graphs. SIKE has

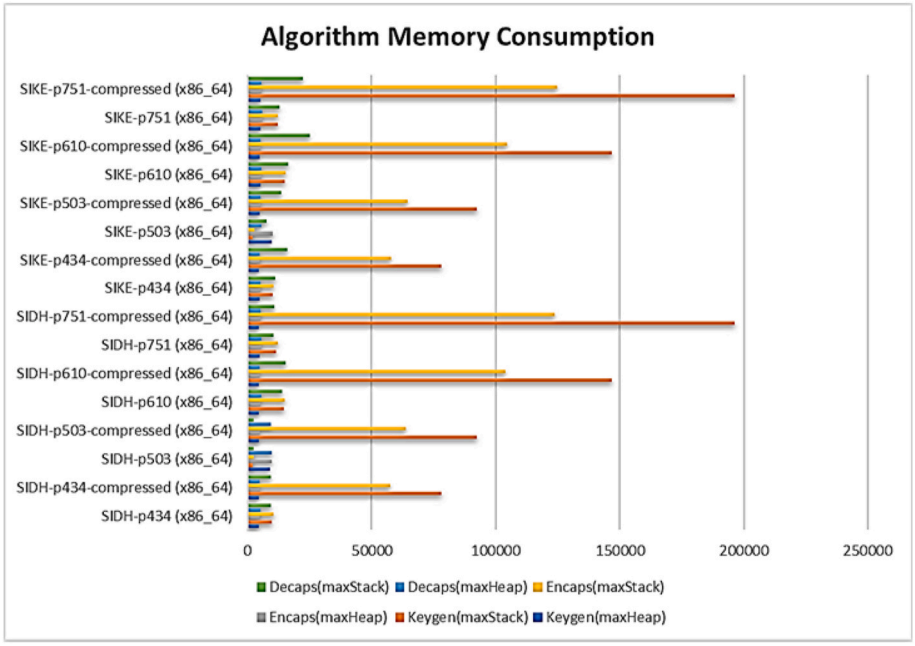

Fig. 23. Memory consumption (bytes per algorithm) (Key Encapsulation Mechanisms using Isogeny Based Algorithms).

Table 14
Lattice-based quantum-safe cryptographic algorithms.

Sl. No.	Algorithm Name	Current Status	Open-Source C Library for Quantum-Safe Cryptographic Algorithms (liboqs)				Usage
			NIST Level	Algorithms Name	Public Key Size (bytes)	Private Key Size (bytes)	
1	CRYSTALS-KYBER	Finalists	1	Kyber512	800	1632	Key Encapsulation Mechanisms
				Kyber512-90s	800	1632	
			3	Kyber768	1184	2400	
				Kyber768-90s	1184	2400	
			5	Kyber1024	1568	3168	
				Kyber1024-90s	1568	3168	
2	NTRU	Finalists	1	NTRU-HPS-2048-509	699	935	Key Encapsulation Mechanisms
			3	NTRU-HPS-2048-677	930	1234	
				NTRU-HRSS-701	1138	1450	
			5	NTRU-HPS-4096-821	1230	1590	
3	SABER	Finalists	1	LightSaber-KEM	672	1568	Key Encapsulation Mechanisms
			3	Saber-KEM	992	2304	
			5	FireSaber-KEM	1312	3040	
4	FrodoKEM	Alternates	1	FrodoKEM-640-AES	9616	19,888	Key Encapsulation Mechanisms
				FrodoKEM-640-SHAKE	9616	19,888	
			3	FrodoKEM-976-AES	15,632	31,296	
				FrodoKEM-976-SHAKE	15,632	31,296	
			5	FrodoKEM-1344-AES	21,520	43,088	
				FrodoKEM-1344-SHAKE	21,520	43,088	
5	NTRU Prime	Alternate	1	ntrulpr653	897	1125	Key Encapsulation Mechanisms
				sntrup653	994	1518	
			2	ntrulpr761	1039	1294	
				sntrup761	1158	1763	
			3	ntrulpr857	1184	1463	
				sntrup857	1322	1999	
6	CRYSTALS-DILITHIUM	Finalists	2	Dilithium2	1312	2528	Digital Signature Algorithms
				Dilithium2-AES	1312	2528	
			3	Dilithium3	1952	4000	
				Dilithium3-AES	1952	4000	
			5	Dilithium5	2592	4864	
				Dilithium5-AES	2592	4864	
7	FALCON	Finalists	1	Falcon-512	897	1281	Digital Signature Algorithms
			5	Falcon-1024	1793	2305	

the smallest public key size but it is also one of the slowest algorithms [10,76,78].

SIKE is the only one algorithm based on Isogney, which could make a place in the NIST 3rd round (Table 11). However, it is declared as an alternate candidate, not the finalist. The runtime behavior and memory consumptions of the algorithm are shown in Table 12, Table 13, and graphically represented in Fig. 20, Fig. 21, Fig. 22, and Fig. 23.

Table 15
Runtime analysis of open quantum safe lattice-based cryptographic algorithms (Key encapsulation mechanisms).

Algorithm	Keygen/s	Keygen (cycles)	Encaps/s	Encaps (cycles)	Decaps/s	Decaps (cycles)
FireSaber-KEM (x86_64)	21419.7	116,577	20332.3	122,807	21,608	115,601
FrodoKEM-1344-AES (x86_64)	589	4,244,465	463.18	5,397,844	475.17	5,261,594
FrodoKEM-1344-SHAKE (x86_64)	200.07	12,494,691	192.33	12,999,407	192.87	12962247
FrodoKEM-640-AES (x86_64)	2179	1147030	1578	1584240	1653	1512386
FrodoKEM-640-SHAKE (x86_64)	794	3,148,801	669.11	3735711	719.76	3473037
FrodoKEM-976-AES (x86_64)	1041.32	2400233	783.74	3189368	816	3,063,901
FrodoKEM-976-SHAKE (x86_64)	371.42	6,731,021	340.22	7348775	341.66	7316407
Kyber1024 (x86_64)	35,936	69,446	31185.3	80,016	41011.33	60,859
Kyber1024-90s (x86_64)	51866.3	48,061	45,639	54,625	67,317	37,038
Kyber512 (x86_64)	67,526	36,882	58398.7	42,674	92577.67	26,910
Kyber512-90s (x86_64)	86,699	28,696	81124.7	30,692	135048.3	18,406
Kyber768 (x86_64)	46470.7	53,657	42638.7	58,493	59629.33	41,825
Kyber768-90s (x86_64)	66,675	37,353	60583.7	41,115	96815.67	25,724
LightSaber-KEM (x86_64)	44989.7	55,426	46,295	53,860	53519.67	46,617
NTRU-HPS-2048-509 (x86_64)	14,699	169,936	60072.3	41,464	71687.33	34,780
NTRU-HPS-2048-677 (x86_64)	8829	283,020	46316.7	53,816	48695.67	51,245
NTRU-HPS-4096-821 (x86_64)	6555.67	381,217	39904.7	62,494	38197.33	65,355
NTRU-HRSS-701 (x86_64)	9552.67	261,573	66261.7	37,568	46262.67	53,946
Saber-KEM (x86_64)	29017.7	86,015	29338.3	85,067	32,348	77,189
ntrulpr653 (x86_64)	32654.3	76,422	29,245	85,325	27985.67	89,232
ntrulpr761 (x86_64)	31496.3	79,234	30106.7	82,886	27409.33	91,111
ntrulpr857 (x86_64)	26,871	92,888	24131.7	103,440	20983.33	119,045
sntrup653 (x86_64)	2980.33	838,768	37,728	66,108	37468.33	66,618
sntrup761 (x86_64)	2349.33	1064025	34414.7	72,484	37,841	65,973
sntrup857 (x86_64)	1814.33	1377979	29393.7	84,894	28335.67	88,127

Table 16
Memory Use Analysis (bytes per algorithm) of Open Quantum Safe Lattice-Based Cryptographic Algorithms (Key Encapsulation Mechanisms).

Algorithm	Keygen (maxHeap)	Keygen (maxStack)	Encaps (maxHeap)	Encaps (maxStack)	Decaps (maxHeap)	Decaps (maxStack)
FireSaber-KEM (x86_64)	8752	25,744	10,256	25,904	10,288	27,408
FrodoKEM-1344-AES (x86_64)	68,832	88,512	90,448	173,352	90,480	216,616
FrodoKEM-1344-SHAKE (x86_64)	69,616	77,352	91,280	123,720	90,480	165,704
FrodoKEM-640-AES (x86_64)	34,160	43,392	43,464	84,008	43,480	104,616
FrodoKEM-640-SHAKE (x86_64)	34,512	37,896	44,248	58,624	44,264	79,264
FrodoKEM-976-AES (x86_64)	51,152	64,616	67,088	128,064	67,376	159,488
FrodoKEM-976-SHAKE (x86_64)	51,936	55,712	67,704	89,568	67,728	121,024
Kyber1024 (x86_64)	13,560	16,720	11,344	21,944	11,376	23,544
Kyber1024-90s (x86_64)	13,560	17,168	10,512	19,928	10,544	21,528
Kyber512 (x86_64)	11,256	7504	8240	11,704	8272	12,504
Kyber512-90s (x86_64)	11,256	7952	7408	9720	7440	10,520
Kyber768 (x86_64)	12,408	11,600	9712	16,344	9744	17,464
Kyber768-90s (x86_64)	12,408	11,984	8880	14,264	8912	15,384
LightSaber-KEM (x86_64)	6416	14,776	7184	14,936	7216	15,704
NTRU-HPS-2048-509 (x86_64)	5810	35,744	6541	30,816	6573	29,120
NTRU-HPS-2048-677 (x86_64)	6340	50,944	7302	44,160	7334	41,888
NTRU-HPS-4096-821 (x86_64)	6996	64,320	8258	56,288	8290	53,632
NTRU-HRSS-701 (x86_64)	6764	49,136	7934	42,352	7966	42,128
Saber-KEM (x86_64)	7472	19,928	8592	20,088	8624	21,208
ntrulpr653 (x86_64)	6198	26,328	7255	26,552	7287	27,864
ntrulpr761 (x86_64)	6509	27,512	7708	27,672	7740	29,208
ntrulpr857 (x86_64)	6823	34,968	8167	35,064	8199	36,760
sntrup653 (x86_64)	6688	23,064	7617	24,120	7649	24,152
sntrup761 (x86_64)	7097	23,672	8168	24,888	8200	24,984
sntrup857 (x86_64)	7497	30,616	8713	31,960	8745	32,056

6.5. Lattice-based cryptography

Lattice-based cryptography is based on the computational hardness of lattice problems, the basis of which is the shortest vector problem (SVP). Here, we are given an input, lattice represented by an arbitrary basis, and our goal is to output the shortest nonzero vector in it [42,59,67]. Simply put, the goal of the attacker is to find the shortest vector from the origin when given the basis of a lattice. A zero vector doesn't work as an answer. The space requirement of this algorithm is exponential and the best-known algorithm has a running time of $2^{O(n \log n)}$ for an exact solution to SVP or even just an approximation to within poly(n) factors. Lattice-based cryptography provides a promising solution for post-quantum cryptography [60,64]. The algorithm is relatively efficient in implementation and has very strong security proofs based on worst-case hardness. The potential quantum-safe algorithm based on the lattice is as follows:

i) **CRYSTALS-KYBER** encompasses two cryptographic primitives: Kyber, an IND-CCA2-secure key-encapsulation mechanism (KEM); and Dilithium, a strongly EUF-CMA-secure digital signature algorithm. Both the algorithms are based on the hard problem using lattice. The algorithm is computationally fast and has a

Table 17
Runtime analysis of open quantum safe lattice-based cryptographic algorithms (digital signature algorithms).

Algorithm	Keypair/s	Keypair (cycles)	Sign/s	Sign (cycles)	Verify/s	Verify (cycles)
Dilithium2 (x86_64)	26417.7	94,494	10,520	237479	29,132	85721
Dilithium2-AES (x86_64)	44800	55671	14407.7	173,353	44537.33	56039
Dilithium3 (x86_64)	16110.3	155032	6506.33	384098	17561.33	142,256
Dilithium3-AES (x86_64)	29631	84225	9637.67	259239	30018.33	83,182
Dilithium5 (x86_64)	10141.7	246360	5279.67	473377	10341	241,650
Dilithium5-AES (x86_64)	19582	127496	8141.33	306924	19511.33	128,032
Falcon-1024 (x86_64)	40.23	62157513	1446.52	1727907	9782.67	255449
Falcon-512 (x86_64)	105.05	23,797,285	2890.33	864862	19684.33	126886

Table 18
Memory Use Analysis (bytes per algorithm) of Open Quantum Safe Lattice-Based Cryptographic Algorithms (Digital Signature Algorithms).

Algorithm	Keygen (maxHeap)	Keygen (maxStack)	Sign (maxHeap)	Sign (maxStack)	Verify (maxHeap)	Verify (maxStack)
Dilithium2 (x86_64)	12656	20,880	11,584	49,616	11360	21592
Dilithium2-AES (x86_64)	12,656	17,040	15,400	44,816	10,528	14,800
Dilithium3 (x86_64)	14,768	25,656	14,569	78,808	14,345	24,728
Dilithium3-AES (x86_64)	14,768	20,792	18,385	69,424	13,513	16,848
Dilithium5 (x86_64)	16,272	33,848	17,375	113,232	17,151	31,128
Dilithium5-AES (x86_64)	16,272	26,936	21,191	108,336	16,319	21,264
Falcon-1024 (x86_64)	12,914	36,792	14,344	80,880	9696	10,512
Falcon-512 (x86_64)	10,994	19,216	11,784	40,944	11,784	2320

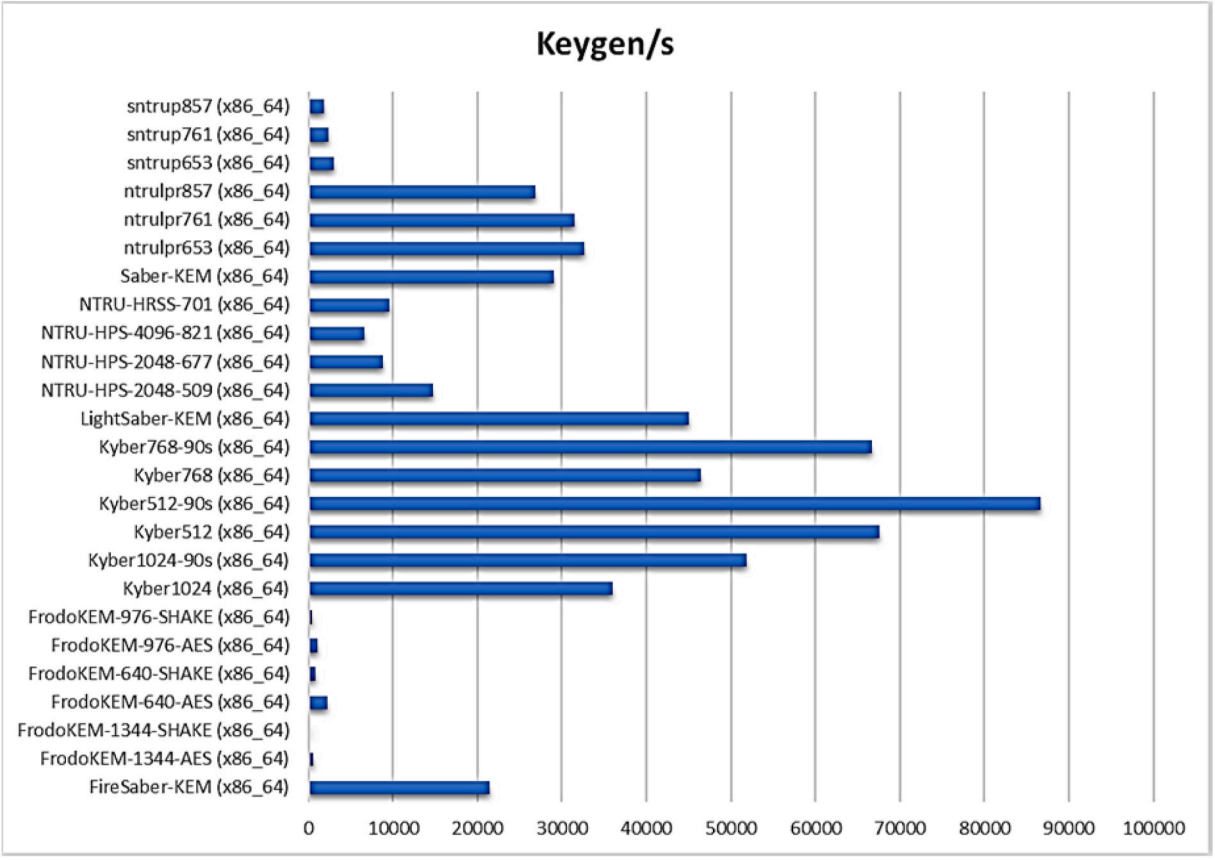

Fig. 24. Keygen operations per second per algorithm (Key Encapsulation Mechanisms using Lattice Based Algorithms).

small public key [32]. It has different parameters set to match the NIST security levels 1, 3, and 5.

ii) **NTRU** is lattice-based cryptography. It was developed in 1996 by mathematicians Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. In 2013, Damien Stehle and Ron Steinfeld developed a post-quantum secure version of NTRU. It is based on the hardness of solving the Ring-learning-with-errors problem (Ring-LWE).

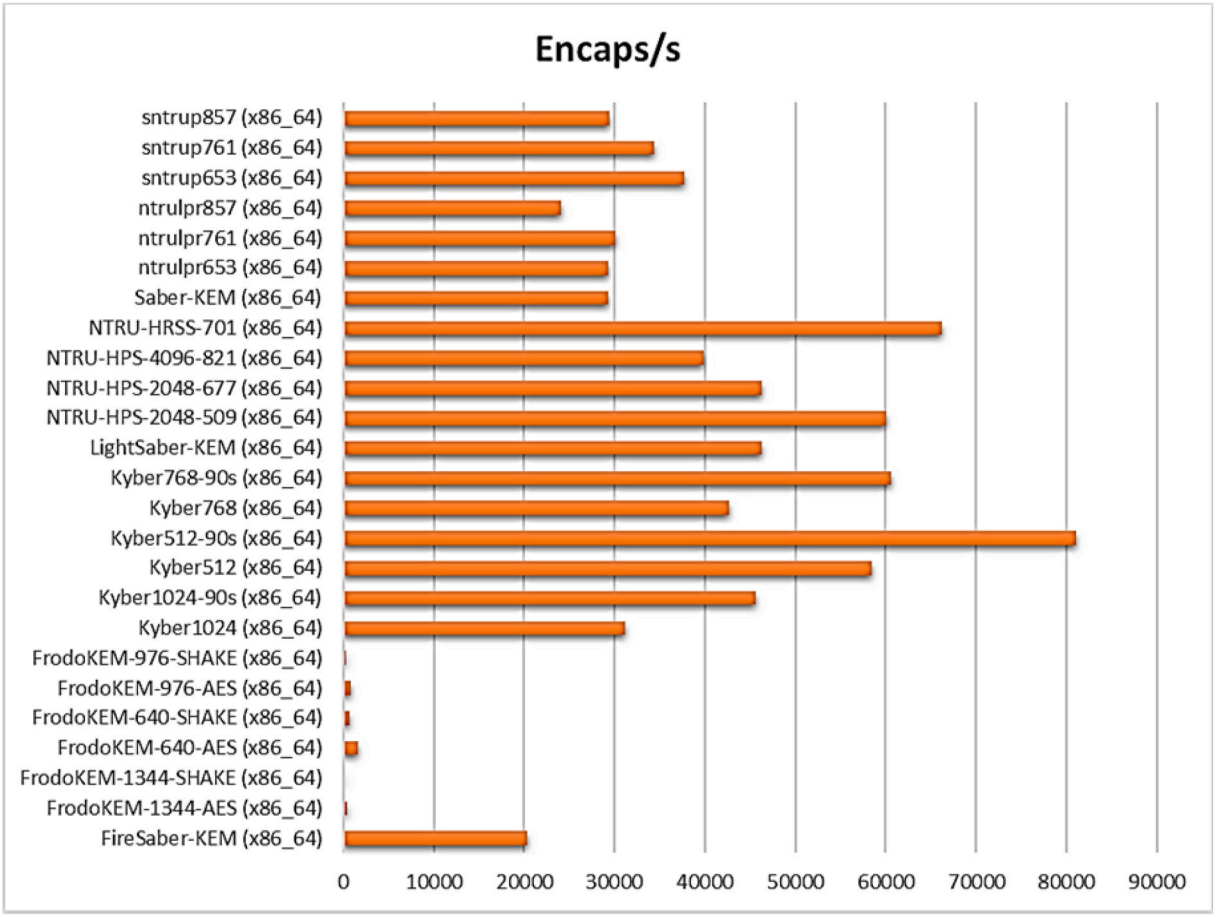

Fig. 25. Encaps operations per second per algorithm (Key Encapsulation Mechanisms using Lattice Based Algorithms).

The algorithm is resistant to attacks using Shor's algorithm [59, 89].

iii) **SABER** also belongs to the lattice-based cryptography algorithm family. The security of the algorithm relies on the hardness of the Module Learning with Rounding problem (MLWR). SABER is a simple and efficient algorithm [34,74]. The SABER-suite offers three security levels:

LightSABER (NIST Level 1)
SABER (NIST Level 3)
FireSABER (NIST Level 5)

iv) **FrodoKEM** is based on the algebraically unstructured lattice. The algorithm has fewer parameter restrictions but a large public key size [43]. FrodoKEM is designed for IND-CCA security at three levels:

FrodoKEM-640, which targets Level 1 in the NIST
FrodoKEM-976, which targets Level 3 in the NIST
FrodoKEM-1344, which targets Level 5 in the NIST

v) **NTRU Prime** is an improvised version of the original NTRU which has eliminated many of the complications of the lattice security review [69].

vi) **CRYSTALS-DILITHIUM** is based on the "Fiat-Shamir with Aborts" technique of Lyubashevsky which uses rejection sampling to make lattice-based Fiat-Shamir schemes compact and secure [36,38]. Dilithium claims to have the smallest public key and signature size of any lattice-based signature scheme that only uses uniform sampling.

vii) **FALCON** is a lattice-based signature scheme designed on the theoretical framework of Gentry, Peikert, and Vaikuntanathan. The security of the algorithm is based on the underlying hard problem of the short integer solution problem (SIS) over NTRU lattices, for which no efficient solving algorithm is currently known in the general case, even with the help of quantum computers [39].

Lattice-based cryptography algorithms seem to be the most promising and quantum-safe. The maximum number of algorithms announced by NIST in 3rd round belongs to the lattice-based cryptography family (Table 14). There are three algorithms' CRYSTALS-KYBER, NTRU and SABER declared as a finalist for Key Encapsulation Mechanisms [54]. CRYSTALS-DILITHIUM and FALCON are declared as a finalist for digital signatures. FrodoKEM and NTRU Prime have been announced as alternate candidates for Key Encapsulation Mechanisms. The runtime behavior and memory consumptions of the algorithms are shown in Table 15, Table 16, Table 17, and Table 18. The data from Table 15 to Table 18 has been graphically represented in Figs. 24–31.

7. Implementation feasibility and migration challenges

Performance analysis of various quantum-safe algorithms shows that in general, quantum-safe algorithms require many CPU cycles for basic operations (i.e., key generation, encrypt/decrypt, key exchange, sign, verify, etc.). The algorithms also require large to a very large public key and private key sizes. The runtime memory consumption of these algorithms is very high compared to the classical cryptographic algorithms.

The higher CPU cycle and memory usage is constrained to the ICT systems and devices. The resource constraints can be resolved for the systems like a laptop, desktop, or high-end server up to some extent. However, it is a challenging issue for small devices like smartphones, sensor networks, smart-grid, IoT, smart devices, smart homes, etc.

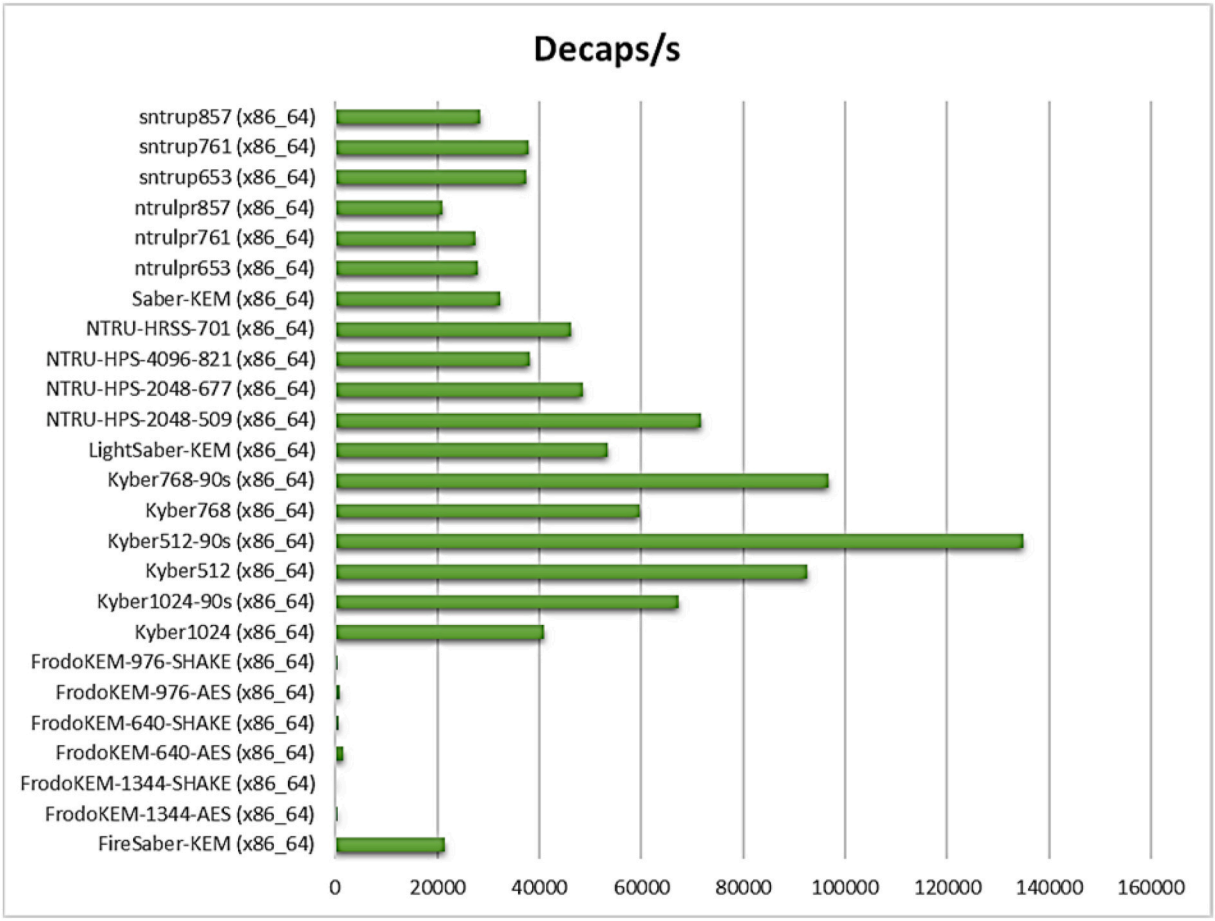

Fig. 26. Decaps operations per second per algorithm (Key Encapsulation Mechanisms using Lattice Based Algorithms).

Security is paramount for all types of ICT devices. Hence the implementation feasibility of quantum-safe algorithms is very important. NIST completed the 3rd round of the process for PQC Standardization in July 2022. Considering various performance and feasibility factors, NIST has selected only four algorithms for standardization (Table 19). Additional four algorithms have been considered for advancement and 4th round of evaluation. The selected algorithm under consideration for the advancement and further evaluation in the 4th round needs to be upgraded on various performance factors to satisfy the implementation feasibility.

Adoption and implementation of these algorithms will depend on various factors such as throughput, latency threshold, key size limitations of current hardware and software, etc. The replacement of classical cryptography algorithms with quantum-safe algorithms will require changes in existing protocol, communication devices, and hardware with high processing capacity to accelerate the algorithm performance, application code, operating system, etc.

NIST is still in the process of finalizing Quantum-Safe algorithms. Most likely the process will continue to the fourth round of evaluation. The algorithms which are declared as a finalist has acceptable performance but require additional analysis to gain sufficient confidence. Some of the algorithms which are declared as alternate may potentially be standardized based on the high confidence in their security.

Currently, the prime objective is to develop a quantum-resistant algorithm to protect from a possible future attack. However, major implementation feasibility issues which require intense effort to overcome are:

- **Migration Overhead and Complexity:** Migration to PQC will require significant upgradation in any organization's current cryptographic infrastructure. Some of their current IT components may become completely obsolete and will require replacement. The existing algorithms, software, and protocol may require redesign and changes. Overall, the entire migration process will add substantial budget overhead to the organization and unavoidable complexity.
- **Scalability Issue:** Scalability is a challenging issue for most of the potential PQC algorithms. It is difficult to prove the hardness of the algorithm at scale. For example, lattice-based cryptography which is one of the popular techniques used for designing quantum-safe algorithms does manage to scale well but achieves average-case hardness. There is a trade-off between hardness and scalability. Either can be achieved, but not both.
- **Large Key Size and Speed:** Most of the potential PQC algorithms require large to very large key sizes compared to the existing cryptography algorithms. Because of the large key size and complex algorithm, more time is required to encrypt, decrypt or verify the message at either end. It also requires more storage space in the device and a longer time to establish secure communication.
- **Resource-Constrained Devices:** Smartphones, IoT Devices, Sensor Networks, Industrial Control Systems, and SCADA are some of the most critical components of our infrastructure. In general, these devices have less memory and processing capacity which poses a challenge for strong security implementation. Post-quantum cryptography algorithms require intense customization and feasibility study for their implementation in the resource-constrained device. It is tough to maintain the hardness of the cryptography algorithm while satisfying the resource constraints.

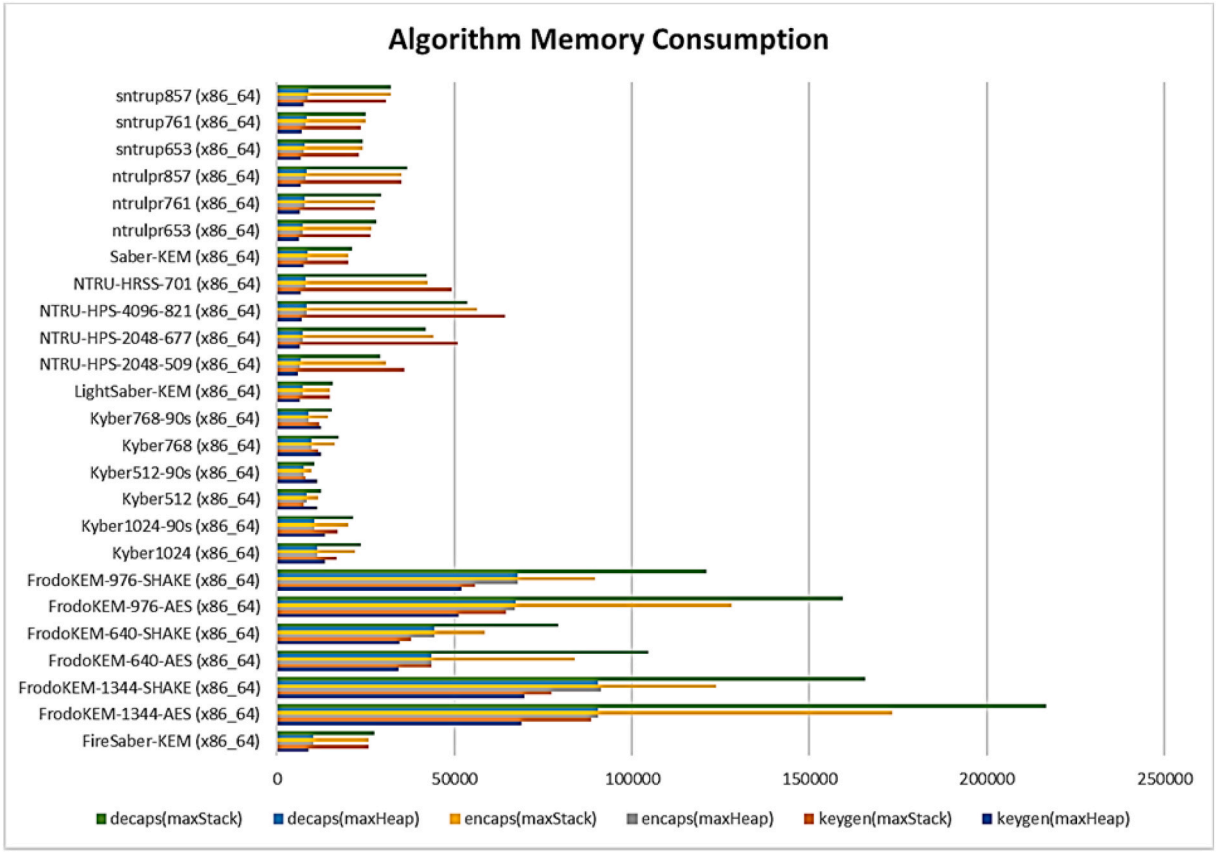

Fig. 27. Memory consumption (bytes per algorithm) (Key Encapsulation Mechanisms using Lattice Based Algorithms).

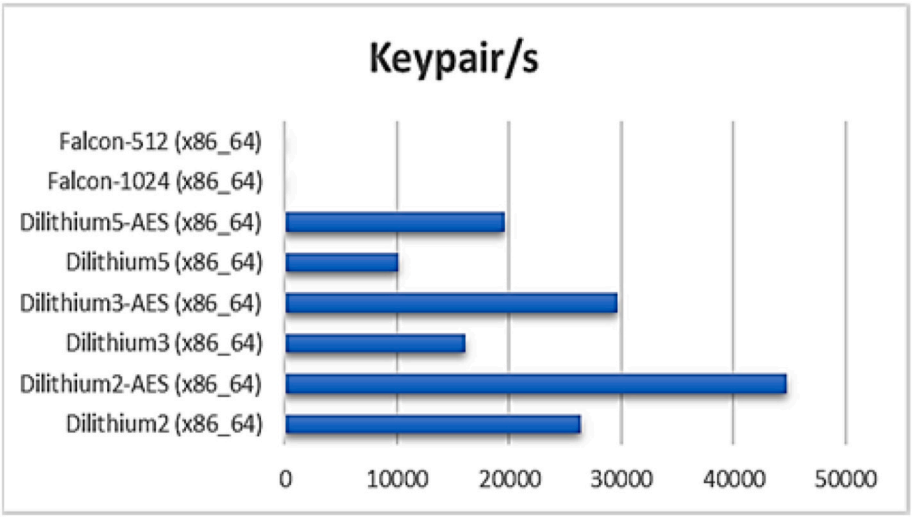

Fig. 28. Keypair operations per second per algorithm (Digital Signature using Lattice Based Algorithms).

8. Conclusions

To break the current cryptographic algorithms, a quantum computer with millions of 'Qubits' is required. It is not sure when quantum computers with a scale of a million qubit will be available in the commonplace. However, the exponential growth in quantum computer technology's development shows that the storm is approaching very fast. The information we communicate today is mostly encrypted using a public-key cryptography algorithm that is vulnerable to quantum computers. It is quite possible that many hackers, state-sponsored or nation-states themselves might be intercepting and harvesting the encrypted messages with envisioning that they will be able to decrypt these messages when the quantum computing resources will be available in the future. To counter this threat, we need to migrate to post-quantum cryptographic algorithms at the earliest. Migration to post-quantum cryptographic algorithms requires thorough planning and implementation globally. The migration process will require work on the various front including education, training, and awareness. It will require extensive coordination among many stakeholders involved in the design and development of hardware, software, and IT infrastructure

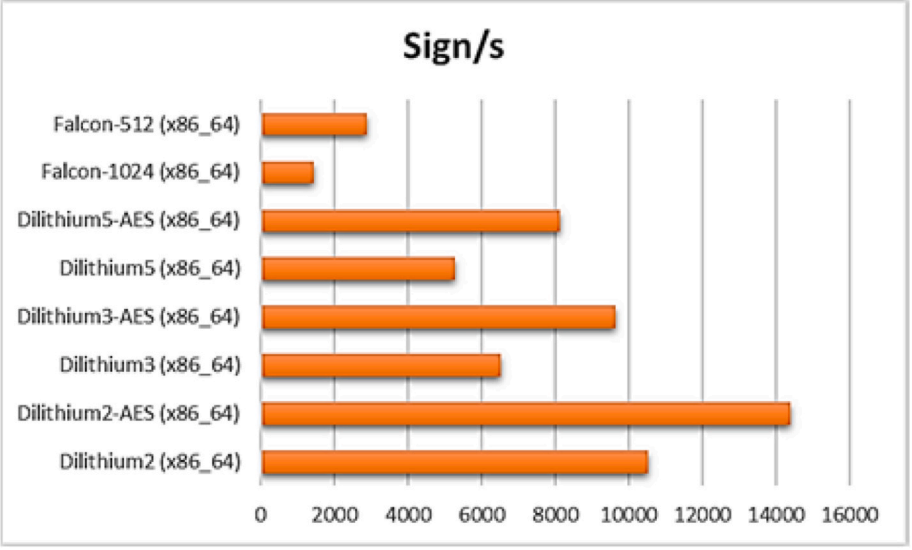

Fig. 29. Sign operations per second per algorithm (Digital Signature using Lattice Based Algorithms).

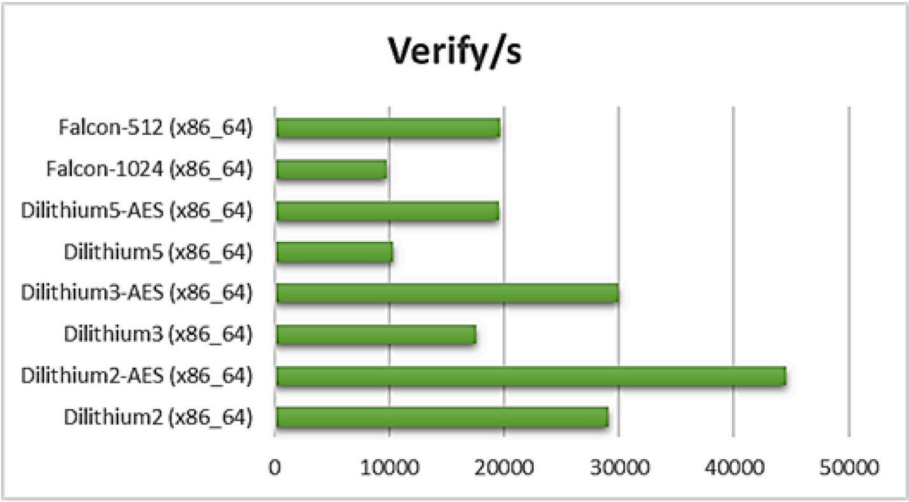

Fig. 30. Verify operations per second per algorithm (Digital Signature using Lattice Based Algorithms).

components. Early preparation will ensure a less costly and effective changeover with minimal disruption.

Funding

Not applicable.

Availability of data and material (data transparency)

Data sharing is not applicable to this article as no new data were created or analyzed in this study.

Code availability

Not applicable.

Ethics approval

Not applicable.

Consent to participate

Not applicable.

Consent for publication

Not applicable.

Compliance with ethical standards

This article does not contain any studies with human participants or animals performed by any of the authors.

Credit author statement

Manish Kumar Conceptualization; Methodology / Study design; Software; Validation; Formal analysis; Investigation; Resources; Data curation; Writing – original draft; Writing – review & editing; Visualization; Supervision; Project administration; Funding acquisition.

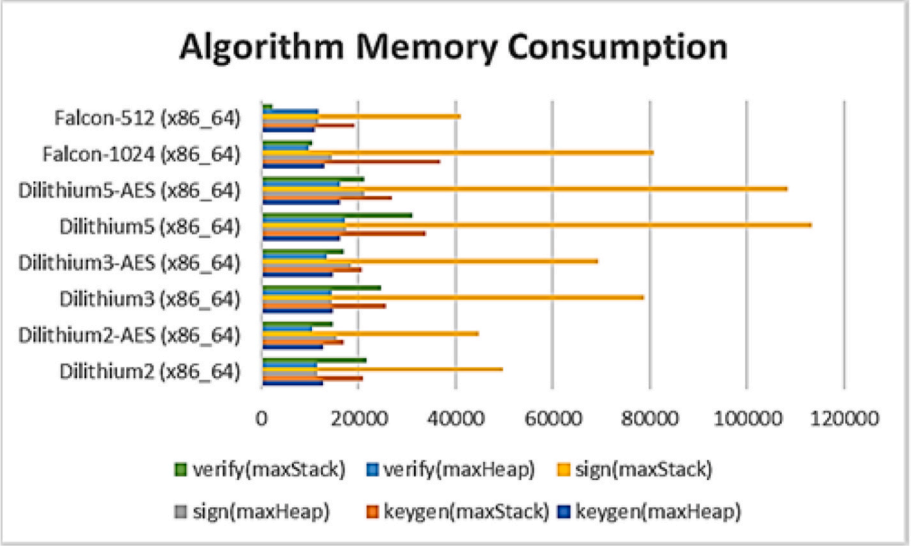

Fig. 31. Memory consumption (bytes per algorithm) (Digital Signature using Lattice Based Algorithms).

Table 19
List of Algorithms selected by NIST for standardization and 4th Round Consideration [3,6].

Algorithms Name	Design	Security	Performance	Overall Assessment
KEM Selected for Standardization				
CRYSTALS-Kyber	Initially, it was IND-CPA-secure PKE scheme, then boosted to an IND-CCA-secure KEM	Strong theoretical security foundation based on lattice cryptography	The performance of KYBER in many different environments is satisfactory	Overall performance of KYBER in software, hardware, and hybrid settings are excellent.
KEMs Advancing to the 4th Round				
BIKE	Based on binary linear quasi-cyclic moderate density parity check (QC-MDPC) codes	Depends on the difficulty of solving the decisional Quasi-cyclic Syndrome Decoding (QCSD) and the decisional Quasi-cyclic Codeword Finding (QCCF) problems	The performance of BIKE would be suitable for most of the applications as confirmed by several hardware benchmarks	Because of the overall performance of the BIKE, it remains under consideration
Classic McEliece	Code Based KEM (Uses Binary Goppa Code)	Based on the hardness of decoding random codes as well as distinguishing scrambled Goppa codes from random codes.	It has the smallest ciphertext among any of the NIST PQC candidates	NIST is confident about the security of Classic McEliece. Hence it is still under consideration
HQC	Based on QC-MDPC codes and follows an LWE-like encryption protocol	The security relies on the difficulty of the Quasi-cyclic Syndrome Decoding(QCSD) with parity problem	Bandwidth of the HQC exceeds that of BIKE, HQC's key generation but decapsulation only requires a fraction of the kilocycles required by BIKE. HQC is one of the top two alternate KEMs.	The overall performance of the HQC is not optimal but still, it is acceptable.
SIKE	Based on isogenies	The security of SIKE follows from the hardness of finding isogenies between supersingular elliptic curves.	It has relatively low communication costs. However, performance on embedded devices may be an issue because the time to perform a single key encapsulation/decapsulation.	It is relatively new scheme. It seems promising but needs further study.
Signatures Selected for Standardization				
CRYSTALS-Dilithium	Lattice-based digital signature algorithm based on the Fiat-Shamir paradigm	The security is based on decisional Module Learning With Errors (MLWE) assumption, which suffices to show that the public key does not leak any information about the secret key.	It uses pseudorandomness and truncated storage techniques improve the performance. The scheme does not use floating-point arithmetic, which is an advantage.	Highly efficient and relatively simple in implementation.
Falcon	Lattice-based signature scheme utilizing the "hash-and-sign" paradigm	The theoretical security is based on the proof of unforgeability in the QROM, based on the hardness of the SIS Problem over NTRU lattices	The verification process is fast and requires low bandwidth. It is the best choice for some constrained protocol scenarios.	NIST has confidence in its security. Because of its low bandwidth, it is a preferred choice for certain applications.
SPHINCS+	Stateless hash-based signature scheme	The security of the SPHINCS+ relies on the security of the underlying hash functions used	Key generation and verification is much faster than signing	It is selected for standardization because it provides a workable signature scheme whose security seems quite solid and is based on an entirely different set of assumptions than those of our other signature schemes to be standardized.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] A public-key cryptosystem based on algebraic coding theory. JPL Deep Space Network Progress 1978:114–6. Report 42– 44, 1978DSNPR..44..114M. Bibcode.

[2] Alagic G, Alperin-Sheriff J, Apon D, Cooper D, Dang Q, Kelsey J, Liu Y-K, Miller C, Moody D, Peralta R, Perlner R, Robinson A, Smith-Tone D. Status report on the second round of the NIST post-quantum cryptography standardization process. Gaithersburg, MD): National Institute of Standards and Technology; 2020. https://doi.org/10.6028/NIST.IR.8309. NIST Internal Report (NISTIR) 8309.

[3] Alagic G, Apon D, Cooper D, Dang Q, Dang T, Kelsey J, Lichtinger J, Miller C, Moody D, Peralta R, Perlne R, Robinson A, Smith-Tone D, Liu Y-K. Status report on the third round of the NIST post-quantum cryptography standardization process. NIST Publications; 2022. https://doi.org/10.6028/nist.ir.8413.

[4] Algorithms. Open Quantum Safe, openquantumsafe.org/liboqs/algorithms. Accessed 11 June 2021..

[5] Althobaiti, Saud Ohood, Dohler Mischa. Cybersecurity challenges associated with the internet of Things in a post-quantum world. IEEE Access 2020;8:157356–81. https://doi.org/10.1109/access.2020.3019345.

[6] Announcing PQC candidates to be standardized, Plus fourth round candidates | CSRC. 2022, July 5. Retrieved, https://Csrc.Nist.Gov/. [Accessed 6 July 2022]. https://csrc.nist.gov/News/2022/pqc-candidates-to-be-standardized-and-round-4. from.

[7] Arute Frank, et al. Quantum supremacy using a Programmable Superconducting processor. Nature 2019;574(7779):505–10. https://doi.org/10.1038/s41586-019-1666-5.

[8] arXiv. Emerging technology from the. How a Quantum Computer Could Break 2048-Bit RSA Encryption in 8 Hours 2 Apr. 2020. MIT Technology Review, www.technologyreview.com/2019/05/30/65724/how-a-quantum-computer-could-break-2048-bit-rsa-encryption-in-8-hours.

[9] Asif Rameez. Post-quantum cryptosystems for internet-of-Things: a Survey on lattice-based algorithms. IoT 2021;2(1):71–91. https://doi.org/10.3390/iot2010005.

[10] Azarderakhsh Reza. NIST Announces round 3 candidates, SIKE remains – PQSecure technologies. Pqsecurity; 27 July 2020. www.pqsecurity.com/nist-announces-round-3-candidates-sike-remains.

[11] Azarderakhsh Reza. WP Release 2/12/2020 – PQSecure Technologies." PQSecure Technologies 14 Feb. 2020. www.pqsecurity.com/white-paper-release-2-12-2020.

[12] Balamurugan C, Singh K, Ganesan G, Rajarajan M. Post-quantum and code-based cryptography—some prospective research directions. Cryptography 2021;5(4):38. https://doi.org/10.3390/cryptography5040038.

[13] Balamurugan Chithralekha, et al. Code-based post-quantum cryptography. 2021. https://doi.org/10.20944/preprints202104.0734.v1. Preprints.

[14] Baldi Marco, et al. Post-quantum cryptography based on codes: state of the art and open challenges. 2017 AEIT International Annual Conference 2017. https://doi.org/10.23919/aeit.2017.8240549.

[15] Banerjee Utsav, et al. Accelerating post-quantum cryptography using an Energy-efficient TLS Crypto-Processor. 2020 IEEE International Symposium on Circuits and Systems (ISCAS) 2020. https://doi.org/10.1109/iscas45731.2020.9180550.

[16] Barker William, et al. Getting ready for post-quantum cryptography: Exploring challenges associated with adopting and using post-quantum cryptographic algorithms. NIST Cybersecurity White Paper 2021. https://doi.org/10.6028/nist.cswp.04282021.

[17] Bernstein, D. J., Chuengsatiansup, C., Lange, T., & Vredendaal, C. (n.d.). NTRU Prime. https://ntruprime.cr.yp.to/. Retrieved April 21, 2021, from https://ntruprime.cr.yp.to/ntruprime-20160511.pdf.

[18] Bernstein DJ, Hülsing A, Kölbl S, Niederhagen R, Rijneveld J, Schwabe P. The SPHINCS + signature framework. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS '19) 2019:2129–46. https://doi.org/10.1145/3319535.3363229.

[19] Beullens W, D'Anvers J-P, Hülsing AT, Lange T, Panny L, de Saint Guilhem C, Smart NP. Post-Quantum Cryptography: current state and quantum mitigation. ENISA; 2021. https://doi.org/10.2824/92307.

[20] Billet O, Gilbert H. Cryptanalysis of Rainbow. Lect Notes Comput Sci 2006:336–47. https://doi.org/10.1007/11832072_23.

[21] Borges Fabio, et al. A Comparison of security and its performance for key Agreements in post-quantum cryptography. IEEE Access 2020;8:142413–22. https://doi.org/10.1109/access.2020.3013250. Crossref.

[22] Bos J, Costello C, Ducas L, Mironov I, Naehrig M, Nikolaenko V, Raghunathan A, Stebila D. Frodo. Proceedings of the 2016 ACM SIGSAC conference on computer and communications security. 2016. https://doi.org/10.1145/2976749.2978425.

[23] Broadbent Anne, Schaffner Christian. Quantum cryptography beyond quantum key Distribution. Des Codes Cryptogr 2015;78(1):351–82. https://doi.org/10.1007/s10623-015-0157-4.

[24] Buchanan William, Woodward Alan. Will quantum computers Be the end of public key encryption? Journal of Cyber Security Technology 2016;1(1):1–22. https://doi.org/10.1080/23742917.2016.1226650.

[25] Casanova A, Faugère JC, Macario-Rat G, Patarin J, Perret L, Ryckeghem J. GeMSS: a great multivariate Short signature. n.d. Retrieved, https://www-polsys.lip6.fr/. [Accessed 6 August 2021]. https://www-polsys.lip6.fr/Links/NIST/GeMSS_specification.pdf. from.

[26] Chen L. Cryptography standards in quantum time: new wine in an Old Wineskin? IEEE Security & Privacy 2017;15(4):51–7. https://doi.org/10.1109/MSP.2017.3151339.

[27] Chen L, Jordan S, Liu Y-K, Moody D, Peralta R, Perlner R, Smith-Tone D (2016) Report on post-quantum cryptography. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Internal Report (NISTIR) 8105. https://doi.org/10.6028/NIST.IR.8105..

[28] Chen Jiajun. Review on quantum communication and quantum computation. J Phys Conf 2021;1865(2):022008. https://doi.org/10.1088/1742-6596/1865/2/022008. Crossref.

[29] Corporation, Isara. "Math Paths to quantum-safe security: hash-based cryptography." ISARA Corporation, www.isara.com/blog-posts/hash-based-cryptography.html. Accessed 2 Aug. 2021.

[30] Costello Craig. Supersingular isogeny key exchange for Beginners. Lect Notes Comput Sci 2020:21–50. https://doi.org/10.1007/978-3-030-38471-5_2.

[31] Cryptographic algorithm - Glossary | CSRC. (n.d.). https://Csrc.Nist.Gov/Glossary/Term/Cryptographic_algorithm. Retrieved April 5, 2022, from https://csrc.nist.gov/glossary/term/cryptographic_algorithm.

[32] CRYSTALS. *CRYSTALS*, pq-crystals.org. Accessed 5 Nov. 2021.

[33] Bellizia D, et al. Post-quantum cryptography: challenges and Opportunities for Robust and secure HW design. In: 2021 IEEE International Symposium on Defect and fault tolerance in VLSI and Nanotechnology systems (DFT); 2021. p. 1–6. https://doi.org/10.1109/DFT52944.2021.9568301.

[34] D'Anvers J-P, Karmakar A, Sinha Roy S, Vercauteren F. Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM. 2019, March 18. Retrieved, https://eprint.iacr.org/. [Accessed 8 May 2022]. https://eprint.iacr.org/2018/230.pdf. from.

[35] Diffie W, Hellman M. New directions in cryptography. IEEE Trans Inf Theor 1976;22(6):644–54. https://doi.org/10.1109/tit.1976.1055638.

[36] Dilithium.Dilithium, pq-crystals.org/dilithium. Accessed 21 July 2021.

[37] Don Jelle, et al. Security of the Fiat-Shamir Transformation in the quantum random-Oracle model. Advances in Cryptology – CRYPTO 2019 2019:356–83. https://doi.org/10.1007/978-3-030-26951-7_13.

[38] Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., & Stehlé, D. (2018). CRYSTALS-dilithium: a lattice-based digital signature scheme. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2018(1), 238–268. https://doi.org/10.13154/tches.v2018.i1.238-268.

[39] Falcon. Falcon, falcon-sign.info. Accessed 3 June 2021.

[40] Feo Luca De. Mathematics of isogeny based cryptography. 2017, 04062. *ArXiv* abs/1711.

[41] Feynman Richard P. Simulating Physics with computers. Int J Theor Phys 1982;21(6–7):467–88. https://doi.org/10.1007/bf02650179.

[42] Fouque P-A, Hoffstein J, Kirchner P, Lyubashevsky V, Pornin T, Prest T, Ricosset T, Seiler G, Whyte W, Zhang Z. n.d. Falcon: Fast-Fourier Lattice-based Compact Signatures over NTRU 2018. Retrieved, https://www.di.ens.fr/. [Accessed 26 March 2021]. https://www.di.ens.fr/~prest/Publications/falcon.pdf. from

[43] FrodoKEM. FrodoKEM, frodokem.org. Accessed 9 July 2021.

[44] GeMSS: A great multivariate Short signature." GeMSS: A Great Multivariate Short Signature, www-polsys.lip6.fr/Links/NIST/GeMSS.html. Accessed 8 Oct. 2021.

[45] Gibney E. Quantum computer race intensifies as alternative technology gains steam. Nature 2020;587(7834):342–3.

[46] Gisin Nicolas, et al. Quantum cryptography. Rev Mod Phys 2002;74(1):145–95. https://doi.org/10.1103/revmodphys.74.145. Crossref.

[47] Global Risk Institute. A Methodology for Quantum Risk Assessment 8 Nov. 2017. Global Risk Institute, globalriskinstitute.org/publications/3423-2.

[48] Global Risk Institute. Quantum Threat Timeline Report 2020 1 Feb. 2021. Global Risk Institute, globalriskinstitute.org/publications/quantum-threat-timeline-report-2020.

[49] Grover Lov K. A fast quantum mechanical algorithm for database search. In: Proceedings of the Twenty-Eighth Annual ACM Symposium on theory of computing - STOC '96; 1996. https://doi.org/10.1145/237814.237866.

[50] Gupta DS, Ray S, Singh T, Kumari M. Post-quantum lightweight identity-based two-party authenticated key exchange protocol for Internet of Vehicles with probable security. Comput Commun 2022;181:69–79. https://doi.org/10.1016/j.comcom.2021.09.031.

[51] Hamming quasi-cyclic (HQC) third round version. 2020, January 10. Retrieved, https://pqc-hqc.org/. [Accessed 11 May 2021]. https://pqc-hqc.org/doc/hqc-specification_2020-10-01.pdf. from.

[52] Help Net Security. Quantum computers: how to prepare for this great threat to information security. Published, Helpnetsecurity.com. [Accessed 24 April 2021]. https://www.helpnetsecurity.com/2020/11/06/quantum-computers-threat/.

[53] IBM Quantum Composer. n.d. https://Quantum-Computing.Ibm.Com/Composer/Docs/Iqx/Guide/Shors-Algorithm. https://quantum-computing.ibm.com/composer/docs/iqx/guide/shors-algorithm.

[54] Bos J, et al. Crystals - Kyber: A CCA-secure Module-lattice-based KEM. In: 2018 IEEE European Symposium on security and privacy (EuroS&P); 2018. p. 353–67. https://doi.org/10.1109/EuroSP.2018.00032.

[55] Joseph D, Misoczki R, Manzano M, Tricot J, Pinuaga FD, Lacombe O, Leichenauer S, Hidary J, Venables P, Hansen R. Transitioning organizations to post-quantum cryptography. Nature 2022;605(7909):237–43. https://doi.org/10.1038/s41586-022-04623-2.

[56] Kazutoshi Kan, Masashi Une. Recent Trends on Research and Development of Quantum Computers and Standardization of Post-Quantum Cryptography 2 Feb.

2021. IMES Discussion Paper Series 21-E-05, Institute for Monetary and Economic Studies, Bank of Japan.
[57] LaMacchia B. The long road ahead to transition to post-quantum cryptography. Commun ACM 2022;65(1):28–30. https://doi.org/10.1145/3498706.
[58] Lee J, Kim W, Kim S, Kim JH. Post-quantum cryptography Coprocessor for RISC-V CPU core. 2022 International Conference on Electronics, Information, and Communication (ICEIC) 2022:1–2. https://doi.org/10.1109/iceic54506.2022.9748834.
[59] Lei X, Liao X. NTRU-KE: a lattice-based public key exchange protocol. 2013, March 11. Retrieved, https://eprint.iacr.org/. [Accessed 12 April 2021]. from https://eprint.iacr.org/2013/718.pdf.
[60] Liu Zhe, et al. Securing Edge devices in the post-quantum internet of Things using lattice-based cryptography. IEEE Commun Mag 2018;56(2):158–62. https://doi.org/10.1109/mcom.2018.1700330.
[61] Kumar M, Pattnaik P. Post quantum cryptography(PQC) - an overview: (Invited paper). 2020 IEEE High Performance Extreme Computing Conference (HPEC) 2020:1–9. https://doi.org/10.1109/HPEC43674.2020.9286147.
[62] Malina, Lukas, et al. "Post-quantum Era privacy protection for Intelligent infrastructures." IEEE Access, vol. 9, 2021, pp. 36038–36077. Crossref, doi: 10.1109/access.2021.3062201.
[63] Malina Lukas, Popelova Lucie, et al. On feasibility of post-quantum cryptography on small devices. IFAC-PapersOnLine 2018;51(6):462–7. https://doi.org/10.1016/j.ifacol.2018.07.104. Crossref.
[64] Micciancio Daniele, Regev Oded. Lattice-based cryptography. Post-Quantum Cryptography 2009:147–91. https://doi.org/10.1007/978-3-540-88702-7_5. *Crossref*.
[65] Mosca Michele. Cybersecurity in an Era with quantum computers: will We Be ready? IEEE Security & Privacy 2018;16(5):38–41. https://doi.org/10.1109/msp.2018.3761723.
[66] Sendrier N. Code-based cryptography: state of the art and perspectives. IEEE Security & Privacy 2017;15(4):44–50. https://doi.org/10.1109/MSP.2017.3151345.
[67] Nejatollahi Hamid, et al. Post-quantum lattice-based cryptography implementations. ACM Comput Surv 2019;51(6):1–41. https://doi.org/10.1145/3292548.
[68] Nicolas Aragon, Barreto Paulo, Bettaieb Slim, Bidoux Loïc, Olivier Blazy, et al. BIKE: bit flipping key encapsulation. 2017. ffhal-01671903f.
[69] NTRU prime: Intro.*NTRU prime: Intro*, ntruprime.cr.yp.to. Accessed 21 June 2021.
[70] Overbeck R, Sendrier N. Code-based cryptography. In: Bernstein DJ, Buchmann J, Dahmen E, editors. Post-quantum cryptography. Berlin, Heidelberg: Springer; 2009. https://doi.org/10.1007/978-3-540-88702-7_4.
[71] Ovilla-Martínez Brisbane, et al. FPGA implementation of some second round NIST lightweight cryptography candidates. Electronics 2020;9(11):1940. https://doi.org/10.3390/electronics9111940.
[72] Rainbow. Open Quantum Safe, openquantumsafe.org/liboqs/algorithms/sig/rainbow.html. Accessed 10 Sept. 2021.
[73] Roma, Crystal Andrea, et al. "Energy efficiency analysis of post-quantum cryptographic algorithms." IEEE Access, vol. 9, 2021, pp. 71295–71317. Crossref, doi:10.1109/access.2021.3077843.
[74] SABER, Lwr-Based KEM. SABER: LWR-Based KEM. www.esat.kuleuven.be/cosic/pqcrypto/saber. [Accessed 21 August 2021].
[75] Samuel GreengardCommissioned by CACM Staff. The Scramble for post-quantum cryptography." news | communications of the ACM. 4 Feb. 2021. cacm.acm.org/news/250331-the-scramble-for-post-quantum-cryptography/fulltext.
[76] Seo H, Anastasova M, Jalali A, Azarderakhsh R. Supersingular isogeny key encapsulation (SIKE) round 2 on ARM Cortex-M4. IEEE Trans Comput 2020;70(10):1705–18.
[77] Sevilla J. Assessing the impact of quantum cryptanalysis. 2020, July 22. https://Docs.Google.Com/Document/d/1lFczb2-ACsm51rJt8ljH0eyNhWrRR0YZ1dlLOZP-Zos/Edit#heading=h.Omwb9lyolrg2. https://docs.google.com/document/d/1lFczb2-ACsm51rJt8ljH0eyNhWrRR0YZ1dlLOZP-zos/edit#heading=h.omwb9lyolrg2.
[78] SIKE – Supersingular Isogeny Key Encapsulation. SIKE – Supersingular Isogeny Key Encapsulation 2018. sike.org. [Accessed 15 September 2021].
[79] Singh H. Code based Cryptography: Classic McEliece 2019. arXiv.Org. https://arxiv.org/abs/1907.12754. July 302019.
[80] SPHINCS+. sphincs.org. Accessed 5 July 2021.
[81] Strand M. A Status Update on quantum safe cryptography. 2021 International conference on military communication and information systems (ICMCIS). 2021. https://doi.org/10.1109/icmcis52405.2021.9486413.
[82] Taraskin Oleg, Soukharev Vladimir, Jao David, LeGrow Jason T. Towards isogeny-based Password-authenticated key establishment. J Math Cryptol 2021;15(1):18–30. https://doi.org/10.1515/jmc-2020-0071.
[83] Tmilinovic. Quantum computing timeline by Gartner. Tmilinovic's Blog 18 Jan. 2019. tmilinovic.wordpress.com/2019/01/18/quantum-computing-timeline-by-gartner.
[84] Verma Pramode. Unconditional security through quantum Uncertainty. International Journal of Critical Infrastructure Protection 2017;16:36–8. https://doi.org/10.1016/j.ijcip.2016.09.001.
[85] Wallden Petros, Kashefi Elham. Cyber security in the quantum Era. Communications of the ACM, 1 Apr. 2019 April 2019;62(4):120. cacm.acm.org/magazines/2019/4/235578-cyber-security-in-the-quantum-era/fulltext.
[86] Wallden Petros, Kashefi Elham. Cyber security in the quantum Era. Commun ACM 2019;62(4):120. https://doi.org/10.1145/3241037.
[87] Wang J, Liu L, Lyu S, Wang Z, Zheng M, Lin F, Chen Z, Yin L, Wu X, Ling C. Quantum-safe cryptography: crossroads of coding theory and cryptography. Sci China Inf Sci 2021;65(1). https://doi.org/10.1007/s11432-021-3354-7.
[88] Wang Liu-Jun, et al. Experimental authentication of quantum key Distribution with post-quantum cryptography. Npj Quantum Information 2021;7(1). https://doi.org/10.1038/s41534-021-00400-7.
[89] Zhang Jiang, et al. Tweaking the Asymmetry of asymmetric-key cryptography on lattices: KEMs and signatures of smaller sizes. Lect Notes Comput Sci 2020:37–65. https://doi.org/10.1007/978-3-030-45388-6_2.
[90] Zhou Tianqi, et al. Quantum cryptography for the future internet and the security analysis. Secur Commun Network 2018;2018:1–7. https://doi.org/10.1155/2018/8214619.

Acronyms

ANSI: The American National Standards Institute
AWS: Amazon Web Service
BIKE: Bit Flipping Key Encapsulation
BSI: Federal Office for Information Security, Germany
CFRG: Crypto Forum Research Group
CPU: Central Processing Unit
ECC: Elliptic Curve Cryptography
ETSI: The European Telecommunications Standards Institute
EUF-CMA: Existential Unforgeability Under Chosen Message Attack
GeMSS: A Great Multivariate Short Signature
HBS: Hash-Based Signature
HFE: Hidden Field Equations Cryptosystem
HQC: Hamming Quasi-Cyclic
ICT: Information and Communications Technology
IETF: The Internet Engineering Task Force
IND-CCA: Indistinguishability Under Chosen Ciphertext Attack
IND-CCA1: Indistinguishability Under (Non-Adaptive) Chosen Ciphertext Attack
IND-CPA: Indistinguishability Under Chosen-Plaintext Attack
KEM: Key Encryption Mechanisms
MLWR: Module Learning with Rounding
NIST: The National Institute of Standards and Technology
NP-Complete: Non-Deterministic Polynomial-Time Complete
NP-Hard: Non-Deterministic Polynomial-Time Hardness
OQS: Open Quantum Safe
OTS: One-Time Signature
PKE: Public Key Encryption
PQC: Post Quantum Cryptography
QCCF: Quasi-cyclic Codeword Finding
QC-MDPC: Quasi-Cyclic Moderate Density Parity-Check
QCSD: Quasi-Cyclic Syndrome Decoding
QROM: Quantum Random Oracle Model
QSC: Quantum-Safe Cryptography
RFC: Request for Comment
Ring-LWE: The Ring-Learning-With-Errors
SCADA: Supervisory Control and Data Acquisition
SIKE: Supersingular Isogeny Key Encapsulation
SIS: Short Integer Solution
SSL/TLS: Secure Sockets Layer /Transport Layer Security
SVP: Shortest Vector Problem
XMSS: eXtended Merkle Signature Scheme