Trade-offs in Post-Quantum Cryptography: A Comparative Assessment of BIKE, HQC, and Classic McEliece

Oleksandr Kuznetsov[1,2], Sergey Kandiy[2], Emanuele Frontoni[1,3], and Oleksii Smirnov[4]

[1] *University of Macerata, Via Crescimbeni, 30/32, Macerata, 62100, Italy*
[2] *V. N. Karazin Kharkiv National University, 4 Svobody Sq., Kharkiv, 61022, Ukraine*
[3] *Marche Polytechnic University, Via Brecce Bianche 12, Ancona, 60131, Italy*
[4] *Central Ukrainian National Technical University, 8, University Ave, Kropyvnytskyi, 25006, Ukraine*

Abstract

This study investigates the trade-offs inherent in three prominent post-quantum cryptographic algorithms: BIKE, HQC, and Classic McEliece. The evaluation of these algorithms was carried out across three different levels of security (L1, L3, and L5), centered on two crucial aspects: cryptographic size parameters and performance efficiency. Classic McEliece emerged as a space-demanding algorithm with significantly larger key sizes but managed to maintain relatively small ciphertext sizes. Conversely, HQC and BIKE presented smaller key and ciphertext sizes, indicating their potential suitability for applications with strict size constraints. In terms of computational costs, Classic McEliece required substantial resources for key generation, whereas HQC and BIKE exhibited balanced performance profiles. The findings underscore the importance of context-specific considerations when choosing an appropriate post-quantum cryptographic algorithm, highlighting the varying strengths and limitations of the analyzed algorithms.

Keywords

Post-quantum cryptography, BIKE, HQC, classic McEliece, performance efficiency, security levels, comparative analysis, trade-offs.

1. Introduction

The precipitous evolution of quantum computing has caused a seismic shift in the cryptography landscape [1–3]. Classic cryptographic algorithms that form the backbone of modern digital security and data privacy could be compromised by quantum computers' immense computational power. This vulnerability engenders an urgent shift towards post-quantum cryptography (PQC), the exploration of cryptographic algorithms thought to be resistant to quantum computer attacks [4–6].

Prominent among these post-quantum cryptographic tools are HQC (Huge Quasi-Cyclic) [7], BIKE (Bit-flipping Key Encapsulation) [8], and Classic McEliece [9] are algorithms conceived to ensure safety in the forthcoming quantum era. The importance of these algorithms is emphasized by the urgent necessity to equip the current security infrastructure with quantum-resistant solutions. However, despite the imperative need and burgeoning interest in these cryptographic methods, comprehensive comparative studies of their performance, security, and resource usage remain limited [10–12].

Against this backdrop, our work seeks to bridge this research gap by providing a thorough comparative analysis of the HQC, BIKE, and Classic McEliece algorithms [13–17]. Each algorithm's effectiveness, security, and resource utilization are scrutinized under a uniform set of metrics to produce an equitable comparison, aiming to guide their adoption and application. Given the criticality of post-quantum

CQPC-2023: Classic, Quantum, and Post-Quantum Cryptography, August 1, 2023, Kyiv, Ukraine
EMAIL: kuznetsov@karazin.ua (O. Kuznetsov); sergeykandy@gmail.com (S. Kandiy); emanuele.frontoni@unimc.it (E. Frontoni); dr.smirnovoa@gmail.com (O. Smirnov)
ORCID: 0000-0003-2331-6326 (O. Kuznetsov); 0000-0003-0552-8341 (S. Kandiy); 0000-0002-8893-9244 (E. Frontoni); 0000-0001-9543-874X (O. Smirnov)

cryptography in mitigating the potential security vulnerabilities that a quantum future might bring, the findings of this study hold significant implications for the cryptographic research community and industry.

The challenge at hand is not only of academic interest but also of paramount importance to the technological security landscape. Thus, our contribution to the post-quantum cryptographic field is not only anticipated to fill a current research void but also aid in understanding and leveraging these pivotal algorithms' strengths and limitations in the ever-evolving quantum environment. We hope that this rigorous exploration will inform further research, standardization processes, and real-world applications of these cryptographic tools, ultimately propelling our collective stride toward a secure quantum future.

2. NIST PQC Security Levels: Classical and Quantum Attacks

The National Institute of Standards and Technology (NIST) initiated the PQC competition intending to discover and standardize novel cryptographic algorithms that can withstand the computational prowess of both classical and quantum computers [2,4,18]. An essential part of this process was setting distinct levels of security robustness to evaluate the algorithms against both types of attacks. Table 1 presents these security levels as defined by NIST for various cryptographic standards [3,19], highlighting the divergence between classical and quantum attacks [20,21].

Table 1
Classical and quantum security for NIST's levels

NIST level		Classical	Quantum
AES-128	(L1)	128	64
SHA3-256	(L2)	128	85
AES-192	(L3)	192	96
SHA3-384	(L4)	192	128
AES-256	(L5)	256	128

As seen in the table, each NIST level corresponds to a specific cryptographic standard, and each standard is defined by the number of bits of security it offers against both classical and quantum attacks.

Level 1 (L1) associated with AES-128, stipulates 128 bits of security against classical attacks. However, against quantum attacks, the level of security drops to 64 bits. This significant drop is due to the potential power of quantum computers and their ability to solve certain problems faster than classical computers, highlighting the unique challenges posed by quantum cryptography.

Level 2 (L2) aligns with SHA3-256 and maintains the same 128 bits of security against classical attacks. Nevertheless, the quantum security raises slightly to 85 bits, representing the relative resilience of this standard to quantum attacks compared to AES-128.

Similarly, Level 3 (L3), represented by AES-192, offers a more substantial 192 bits of classical security, but the quantum security level, like its Level 1 counterpart, halves to 96 bits.

Level 4 (L4) and Level 5 (L5), corresponding to SHA3-384 and AES-256 respectively, hold the same 192 and 256 bits of classical security, respectively. Notably, the quantum security levels plateau at 128 bits, reflecting the fact that the computational advantage of quantum machines does not indefinitely scale against all forms of encryption.

It's crucial to conceptualize these standards not as definitive thresholds but as guidelines for assessing an algorithm's relative security. Cryptographic strength is not an absolute measure; it is contextual, based on the capabilities of potential adversaries. Therefore, the outlined security levels aim to provide a performance baseline for cryptographic algorithms and set minimum requirements for their strength against potential classical and quantum threats. By appreciating these security levels and the role they play, we can more effectively gauge the resilience of post-quantum cryptographic algorithms, including HQC, BIKE, and Classic McEliece, which are central to this study.

Each of the examined algorithms - HQC, BIKE, and Classic McEliece - is built upon the foundational principle of coding theory. This theory allows them to construct robust and secure key encapsulation and public-key encryption algorithms resistant to post-quantum threats. The design of these algorithms and the choice of their parameters stem from the authors' insights into constructing a secure cryptographic system. The parameters for these algorithms are given at varying levels of security, ranging from 1 to 5, as established by NIST (Table 2).

These parameters include:

- n is the length of the code, which is the number of bits in the codeword,
- k is the length of the information word, i.e., the number of information bits each codeword can hold, and
- t is the error-correcting capacity, defining the maximum number of errors that the code can correct.

The Classic McEliece algorithm employs Goppa codes. As we move from security level 1 to 5, the parameters for Classic McEliece increase. Specifically, the codeword length n rises from 3488 to 8192, the information word length k enhances from 2720 to 6528, and the error-correcting capacity t augments from 64 to 128. This increment underpins an enhanced level of security, attributable to a larger key size.

BIKE leverages low-density parity-check (LDPC) quasi-cyclic codes and bit-flipping decoding. Its parameters also see a significant rise from security level 1 to level 5. The code length n escalates from 24646 to 81946, the length of the information word k inflates from 12323 to 40973, and the error-correcting capacity for both messages and keys enhances from 134 to 264 and 142 to 274, respectively. These augmentations represent an increase in the robustness of the algorithm against potential attacks.

HQC utilizes quasi-cyclic codes similar to BIKE but with broader applicability in a range of cryptographic protocols. The parameters for HQC also grow from security level 1 to level 5. The code length n expands from 35338 at level 1 to 115274 at level 5. The information word length k swells from 17669 to 57637, and the error-correcting capacity t rises from 132 to 262. These escalating parameters reflect the adaptability of HQC to various security levels.

These parameter increments from security level 1 to 5 reflect an increase in key size and, therefore, resilience to attacks. The higher the security level, the more computational resources would be required to launch a successful attack on the cryptographic system. Therefore, the choice of security level is a balancing act between security requirements and computational and storage resources.

Table 2 visually demonstrates how these code parameters influence the security level in the three algorithms [20,22]. This visualization provides valuable insights when choosing between them, depending on the specific application requirements. By understanding the correlation between the code parameters and the level of security, one can better select the algorithm that meets their security, performance, and resource needs.

3. Cryptographic and Performance Metrics of HQC, BIKE, and Classic McEliece Algorithms

3.1. Code-based Cryptosystems: Advantages and Limitations

Code-based cryptography represents a subclass of post-quantum cryptographic systems that utilizes the principles of error-correcting codes to achieve security against quantum computer attacks [23–27]. Pioneered by Robert McEliece in 1978 with the introduction of the McEliece cryptosystem [28], code-based cryptography has since evolved, fostering a rich field of research and development [29–31].

At the core of code-based cryptography lies the principle of error-correcting codes—specifically, the mathematical challenge of decoding a general linear code, known as the 'decoding problem’. The security of a code-based cryptosystem fundamentally relies on the computational hardness of this decoding problem. If an adversary intercepts the ciphertext, they would need to solve the decoding problem to retrieve the original plaintext [23–27].

Code-based cryptosystems operate through three primary processes: key generation, encryption, and decryption. During key generation, a public/secret key pair is produced, where the public key is a purposely flawed error-correcting code and the secret key is the corresponding unflawed code. The encryption process involves embedding the message into a codeword and introducing specific errors, which are then corrected during the decryption phase using the secret key.

Prominent examples of code-based cryptographic systems include the original McEliece cryptosystem, its derivative Niederreiter cryptosystem, and more contemporary entrants such as BIKE and HQC, which introduce advanced error-correction code strategies, offering robust security and performance trade-offs.

Advantages:

- Quantum-Resistance. Code-based cryptography's principal advantage is its resilience against quantum computer attacks. As the security of these systems relies on the difficulty of the decoding problem, they remain secure even against Shor's algorithm – the most powerful known quantum algorithm for factoring integers and computing discrete logarithms in polynomial time.

- Maturity and Robustness. The McEliece cryptosystem, the bedrock of code-based cryptography, has withstood the test of time, remaining unbroken in its original form since its inception in 1978. This longevity underscores the robustness of the underlying mathematical principles of code-based cryptography.
- Efficiency. Code-based cryptosystems generally offer efficient encryption and decryption processes. For instance, the encryption and decryption in the McEliece system only involve matrix multiplication and error correction, respectively, both of which can be efficiently implemented.

Limitations:

- Key Size. The most substantial drawback of many code-based cryptosystems, such as the McEliece and Niederreiter systems, is the large size of the public key. This can limit their applicability in environments with strict bandwidth or storage limitations.
- Structure Leakage. Some code-based cryptosystems that use structured codes to reduce key sizes may leak information about the secret key, potentially compromising their security. This is a delicate balancing act, requiring careful design to prevent structure-related attacks.
- Security Parameter Selection. The selection of appropriate security parameters (e.g., code length, error weight) for code-based cryptosystems requires careful consideration. Insufficient parameters can jeopardize security, while overly conservative parameters can result in inefficiency.

In conclusion, while code-based cryptography presents a compelling approach to achieving quantum resistance, the key challenges lie in navigating the trade-offs between key sizes, security, and performance. As research progresses in this field, promising directions include exploring advanced coding techniques and optimizations to enhance the efficiency and practicability of these cryptosystems. Despite the challenges, the proven resilience and quantum-resistant properties of code-based cryptography affirm its valuable role in the post-quantum cryptography landscape.

3.2. Classic McEliece

The Classic McEliece algorithm’s cryptographic and performance metrics are depicted in Tables 3 and 4, respectively. These metrics provide insights into the cryptographic system's efficiency and security aspects.

The cryptographic parameters highlighted for the Classic McEliece algorithm include the public and private key sizes, the ciphertext size, and the session key size. These metrics are fundamental in understanding the cryptographic overhead of the system and its associated level of security.

The public key size grows substantially from 261120 bytes at NIST level 1 to 1357824 bytes at NIST level 5c. The private key size also sees a significant increment from 6492 bytes at level 1 to 14120 bytes at level 5c. These increases align with the general principle that larger key sizes translate into stronger security, making the system more resilient against cryptographic attacks. The ciphertext size and the session key size also increase as the NIST level progresses, pointing to stronger security and larger communication overheads. However, the session key size remains consistent at 32 bytes, as its primary role is to ensure confidentiality and integrity during a session, regardless of the NIST level.

Table 4 displays the Classic McEliece algorithm's performance measures: KeyGen, Encaps, and Decaps. These metrics measure the computational efficiency of key generation, encapsulation, and decapsulation processes, respectively.

KeyGen is the key generation process involves creating a public and private key pair. As the security level increases, the computational cost also grows substantially, from around 56.7 million cycles at level 1 to about 486.2 million cycles at level 5c.

Encaps is the encapsulation process involves generating a ciphertext and an associated symmetric key. This process also requires more computational cycles as the NIST level increases, going from about 36.5 thousand cycles at level 1 to around 157 thousand cycles at level 5c.

Table 2
Code parameters in HQC, BIKE, and Classic McEliece algorithms

NIST level		1	3	5(a)	5b	5c
Classic McEliece	n	3488	4608	6688	6960	8192
	k	2720	3360	5024	5413	6528
	t	64	96	128	119	128
BIKE	n	24646	49318	81946		
	k	12323	24659	40973		
	t (message)	134	199	264		
	t (key)	142	206	274		
HQC	n	35338	71702	115274		
	k	17669	35851	57637		
	t	132	200	262		

Table 3
Cryptographic parameters of the Classic McEliece algorithm

NIST level	Designation	Public key size, bytes	Private key size, bytes	Ciphertext size, bytes	Session key size, bytes
L1	mceliece348864	261120	6492	96	32
L3	mceliece460896	524160	13608	156	32
L5	mceliece6688128	1044992	13932	208	32
L5b	mceliece6960119	1047319	13948	194	32
L5c	mceliece8192128	1357824	14120	208	32

Table 4
Performance indicators of the Classic McEliece algorithm (AVX512), cycles

NIST level	Designation	Performance Indicators (AVX-enabled), cycles		
		KeyGen	Encaps	Decaps
L1	mceliece348864	56705880	36457	127140
L3	mceliece460896	153266214	76086	263046
L5	mceliece6688128	443746986	171442	306212
L5b	mceliece6960119	316995472	144678	286596
L5c	mceliece8192128	486195290	156945	310097

Table 5
Cryptographic and performance metrics of the BIKE algorithm

NIST Level	Public key, bits	Private key, bits	Ciphertext, bits	Performance Indicators (AVX512-enabled), cycles		
				KeyGen	Encaps	Decaps
L1	12323	2244	12579	589	97	1135
L3	24659	3346	24915	1823	223	3887
L5	40973	4640	41229	—	—	—

Table 6
Cryptographic metrics of the HQC algorithm

NIST Level	Designation	Public key size, bytes	Private key size, bytes	Ciphertext size, bytes
L1	hqc-128	2249	56	4497
L3	hqc-192	4522	64	9042
L5	hqc-256	7245	72	14485

Table 7
Performance Metrics of the HQC Algorithm

NIST Level	Designation	KeyGen, kilocycles	Encaps, kilocycles	Decaps, kilocycles
L1	hqc-128	87	204	362
L3	hqc-192	204	465	755
L5	hqc-256	409	904	1505

Decaps is the decapsulation process that entails recovering the symmetric key from the ciphertext using the private key. Similar to the other processes, its computational cost rises as the security level augments, moving from approximately 127.1 thousand cycles at level 1 to about 310.1 thousand cycles at level 5c.

3.3. BIKE

The cryptographic and performance metrics of the BIKE algorithm are presented in Table 5. These metrics facilitate a comprehensive understanding of the system's security and efficiency characteristics. As the NIST security level increases from L1 to L5, there is a corresponding increase in the size of the public key, private key, and ciphertext. For instance, the size of the public key expands from 12323 bits at L1 to 40973 bits at L5. Similarly, the private key size grows from 2244 bits at L1 to 4640 bits at L5. The ciphertext size also enlarges, from 12579 bits at L1 to 41229 bits at L5. The increased sizes underscore the reinforced security level, albeit at the expense of larger communication overheads.

Table 5 showcases the performance metrics for key generation (KeyGen), encapsulation (Encaps), and decapsulation (Decaps) processes of the BIKE algorithm. As the security level escalates from L1 to L3, the computational cost for these processes also increases. For instance, the KeyGen process escalates from 589 kilocycles at L1 to 1823 kilocycles at L3. Similarly, the Encaps and Decaps processes see an increase in computational cost from L1 to L3. Performance metrics for L5 are absent, potentially due to computational constraints or the metrics were unavailable at the time of the report.

In summary, similar to the Classic McEliece, the BIKE algorithm presents a trade-off between security and computational efficiency. Increased security levels lead to larger key sizes and ciphertexts, as well as increased computational costs. Selecting an appropriate NIST level depends on balancing the need for security and the available computational and storage resources.

3.4. HQC

The cryptographic characteristics and performance measures of the HQC algorithm are elucidated in Tables 6 and 7 respectively. This comprehensive data enables us to gauge the algorithm's balance between security and efficiency.

In Table 6, we observe that as the NIST security level increases from L1 to L5, there is a corresponding augmentation in the size of the public key, private key, and ciphertext. The public key size, for instance, expands nearly three-fold from 2249 bytes at L1 to 7245 bytes at L5. The private key size sees a smaller expansion, from 56 bytes at L1 to 72 bytes at L5. The ciphertext size also escalates significantly from 4497 bytes at L1 to 14485 bytes at L5. These size increases depict the enhanced security level, although they may necessitate larger communication and storage overheads.

Table 7 details the performance metrics associated with key generation (KeyGen), encapsulation (Encaps), and decapsulation (Decaps) processes for the HQC algorithm. As the security level progresses from L1 to L5, the computational costs for these processes exhibit a clear upward trend. The KeyGen process, for example, escalates from 87 kilocycles at L1 to 409 kilocycles at L5. Similarly, the computational costs for Encaps and Decaps processes also increase from L1 to L5.

In summary, the HQC algorithm, like the Classic McEliece and BIKE algorithms, showcases a trade-off between security and computational efficiency. The choice of NIST level depends on the balance between security needs and computational/storage resources available.

4. Comparative Analysis Algorithms

To compare the cryptographic characteristics of HQC, BIKE, and Classic McEliece algorithms, we examine Tables 8 to 10 and corresponding figures for various levels of security (L1, L3, L5).

Table 8
Comparative analysis of cryptographic metrics at security level L1

Algorithm	Public key size, bytes	Private key size, bytes	Ciphertext size, bytes
HQC	2249	56	4497
BIKE	1541	281	1573
Classic McEliece	261120	6492	96

Table 9
Comparative analysis of cryptographic metrics at security level L3

Algorithm	Public key size, bytes	Private key size, bytes	Ciphertext size, bytes
HQC	4522	64	9042
BIKE	3083	419	3115
Classic McEliece	524160	13608	156

Table 10
Comparative analysis of cryptographic metrics at security level L5

Algorithm	Public key size, bytes	Private key size, bytes	Ciphertext size, bytes
HQC	7245	72	14485
BIKE	5122	580	5154
Classic McEliece	1044992	13932	208

These provide a comprehensive overview of the algorithm's performance in terms of key sizes and ciphertext size.

Figs. 1–3 show the corresponding diagrams that visually allow you to compare the relevant indicators.

At security level L1, HQC and BIKE exhibit relatively small sizes for public keys, private keys, and ciphertexts compared to the Classic McEliece algorithm. Classic McEliece features an enormously larger public key size (261120 bytes), which may impose significant storage and communication overheads. Conversely, its ciphertext size is remarkably small (96 bytes), potentially providing benefits in scenarios where ciphertext size is a crucial factor.

As we escalate to security level L3, a similar trend is observable. The Classic McEliece algorithm continues to dominate with a significantly larger public key size (524160 bytes) and concurrently maintains a smaller ciphertext size (156 bytes). HQC and BIKE still exhibit more modest key and ciphertext sizes, which may be advantageous in resource-constrained environments.

At the highest security level L5, Classic McEliece's public key size grows to an astounding 1044992 bytes. Conversely, HQC and BIKE maintain relatively smaller sizes for public and private keys and ciphertexts. This contrast portrays the significant trade-off between security and efficiency across the algorithms.

Given these comparative analyses, it becomes clear that the Classic McEliece algorithm provides robust security with the cost of substantially larger public keys, while HQC and BIKE offer a more balanced profile for key sizes and ciphertext sizes. Ultimately, the choice of an algorithm will rely on the specific requirements of the application, particularly considering the trade-off between security level, storage and computational resources, and communication overhead.

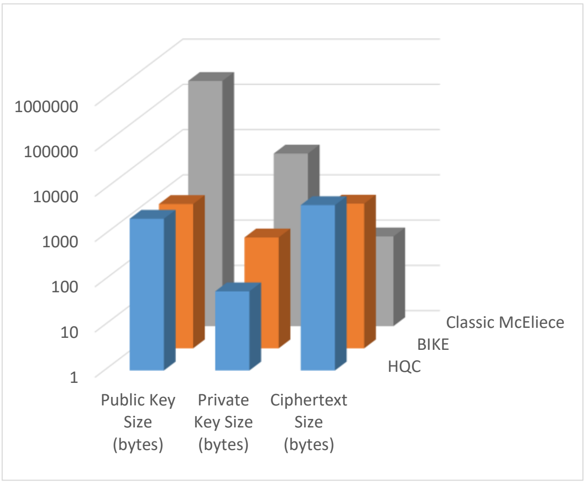

Figure 1: Results of comparing the cryptographic performance for the L1 level

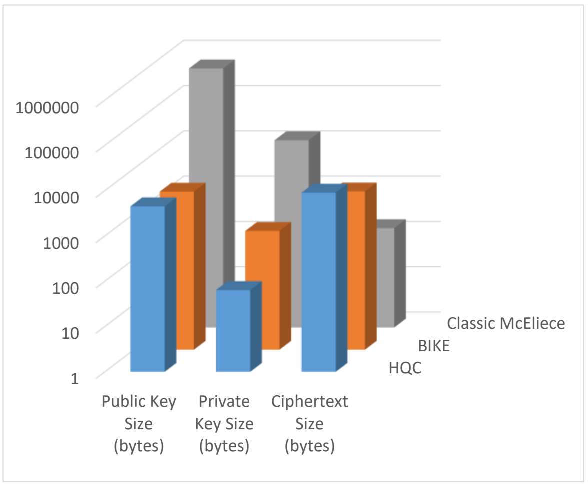

Figure 2: Results of comparing the cryptographic indicators for the L3 level

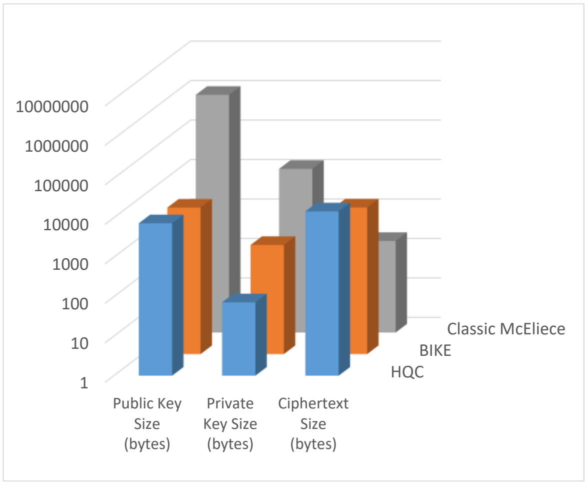

Figure 3: Results of comparing the cryptographic indicators for the L5 level

The efficiency of the cryptographic algorithms is further explored in Tables 11 to 13 and Figs. 4 to 6, presenting a comparison of the performance indicators for different security.

Table 11
Comparative analysis of performance metrics at security level L1 (kilocycles)

Algorithm	KeyGen	Encaps	Decaps
HQC	87	204	362
BIKE	589	97	1135
Classic McEliece	56706	36	127

Table 12
Comparative analysis of performance metrics at security level L3 (kilocycles)

Algorithm	KeyGen	Encaps	Decaps
HQC	204	465	755
BIKE	1823	223	3887
Classic McEliece	153266	76	263

Table 13
Comparative analysis of performance metrics at security level L5 (kilocycles)

Algorithm	KeyGen	Encaps	Decaps
HQC	409	904	1505
BIKE	—	—	—
Classic McEliece	443747	171	306

At the L1 security level, Classic McEliece requires significantly more kilocycles for key generation (56706), but it compensates with low-cost encapsulation and decapsulation procedures. In contrast, HQC exhibits the most efficient key generation, while BIKE shows the lowest encapsulation cost.

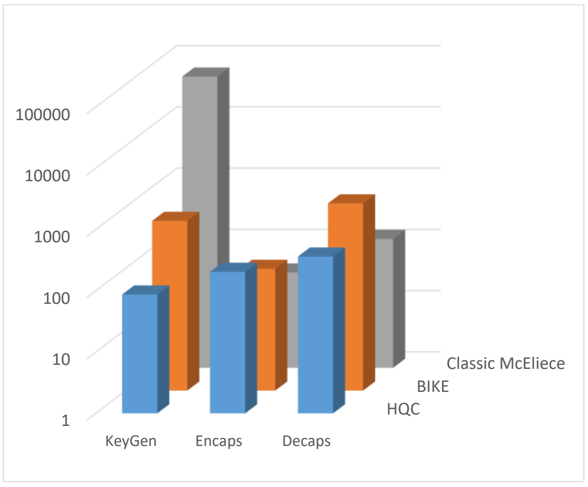

Figure 4: Performance comparison results for L1 stability level (kilocycles)

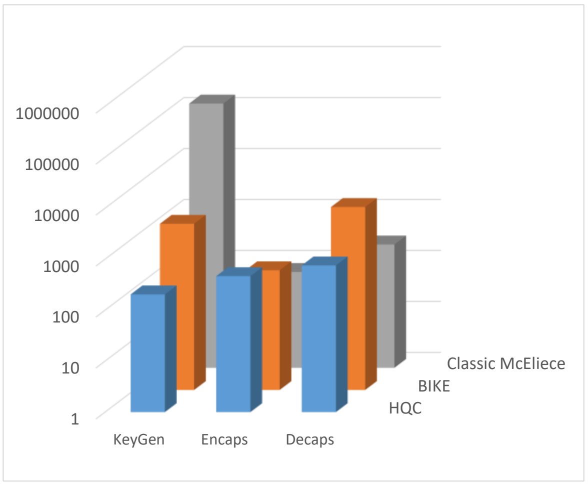

Figure 5: Performance comparison results for L3 stability level (kilocycles)

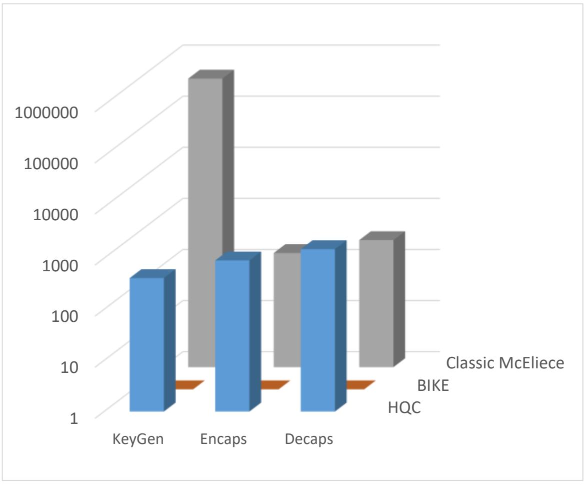

Figure 6: Performance comparison results for L5 stability level (kilocycles)

As we move to security level L3, the trend continues: Classic McEliece consumes the most computational resources for key generation, while maintaining relatively low costs for encapsulation and decapsulation. Again, HQC proves the most efficient for key generation, while BIKE requires fewer kilocycles for encapsulation.

At the highest security level (L5), Classic McEliece's computational costs for key generation soar to 443747 kilocycles, maintaining its tendency towards efficiency in encapsulation and decapsulation. HQC remains steady with relative efficiencies in all three performance metrics. Unfortunately, performance data for BIKE at this security level is missing.

Based on these comparative analyses, it is evident that while Classic McEliece demands a significant computational investment for key generation, it provides efficiency in encapsulation and decapsulation. Conversely, HQC and BIKE generally present a more balanced computational profile across key generation, encapsulation, and decapsulation processes. However, the absence of data for BIKE at the L5 security level makes it challenging to draw comprehensive conclusions. Again, the choice of algorithm would rely on specific application requirements, including trade-offs between security level, computational resources, and performance efficiency.

5. Discussion

The evaluation of post-quantum cryptographic algorithms BIKE, HQC, and Classic McEliece showcased distinctive attributes for each regarding their cryptographic size parameters and performance efficiencies across three levels of security (L1, L3, L5). The investigation has shed light on the significant trade-offs inherent in the adoption of these algorithms, primarily concerning computational efficiency, key, and ciphertext size, and the level of security provided.

Among the considered algorithms, Classic McEliece showed the most substantial key sizes, regardless of the security level. It emerged as the most space-demanding algorithm, with public keys ranging from approximately 261KB at L1 to over 1MB at L5. This substantial key size can pose issues for storage and transmission, making it potentially less suitable for constrained environments such as IoT devices. Yet, it was observed that Classic McEliece manages to maintain relatively small ciphertext sizes, especially at lower security levels.

On the other hand, HQC and BIKE demonstrated smaller key and ciphertext sizes across all security levels, potentially making them more appropriate for applications with strict size constraints. However, BIKE's performance metrics at the L5 security level were not available, which restricts the full understanding of its capabilities and limitations at this higher level of security.

As for performance efficiency in terms of computational costs, Classic McEliece required significantly more computational resources for a key generation across all security levels. This aspect might limit its adoption in environments where computational power is a primary concern, despite its efficiency in the encapsulation and decapsulation processes. Meanwhile, HQC demonstrated an overall balanced performance profile with relative efficiencies across all procedures. BIKE, except for the missing data at L5, also indicated a good balance between key generation, encapsulation, and decapsulation.

It is crucial to note that the choice of algorithm would ultimately rely on the specific application context and its requirements. For instance, in scenarios where computational resources and storage are not stringent, Classic McEliece might be an appropriate choice due to its relative performance efficiency. Conversely, in situations with strict size limitations, HQC and BIKE might be the more suitable algorithms.

In conclusion, this comparative analysis provides valuable insights into the properties and performance trade-offs of BIKE, HQC, and Classic McEliece, potentially assisting practitioners in selecting the appropriate post-quantum cryptographic algorithm based on their particular requirements. However, it also underscores the need for more comprehensive and comparative studies to better understand these algorithms' potential and challenges, especially at higher security levels.

6. Conclusions

Our investigation of the BIKE, HQC, and Classic McEliece post-quantum cryptographic algorithms revealed distinct characteristics and trade-offs for each, primarily in the areas of cryptographic size parameters and performance efficiencies. The analysis underscored the significance of application context and specific requirements when selecting an appropriate cryptographic algorithm.

Classic McEliece, despite its large key sizes, displayed relatively small ciphertext sizes and efficient encapsulation and decapsulation performance. These properties suggest that Classic McEliece could be a suitable choice in contexts where computational power and storage are not significant constraints. On the other hand, BIKE and HQC demonstrated a more balanced profile in terms of size parameters and performance metrics, indicating their potential suitability for applications with stricter size limitations. However, the lack of BIKE performance data at the L5 security level calls for further investigation to fully comprehend its potential and limitations at this higher level of security.

This comparative analysis provides a robust foundation for practitioners when choosing a post-quantum cryptographic algorithm tailored to their particular requirements. It further emphasizes the need for continued, comprehensive comparative studies to fully appreciate the potential and challenges of these post-quantum cryptographic algorithms, particularly at higher security levels.

7. Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No. 101007820. This publication reflects only the author's view and the REA is not responsible for any use that may be made of the information it contains.

8. References

[1] N. Koblitz, A.J. Menezes, A Riddle Wrapped in an Enigma (2015). URL: http://eprint.iacr.org/2015/1018

[2] L. Chen, et al., Report on Post-Quantum Cryptography, National Institute of Standards and Technology (2016). doi: 10.6028/nist.ir.8105

[3] I.T.L. Computer Security Division, Post-Quantum Cryptography: Proposed Requirements & Eval Criteria, CSRC, NIST (2016). URL: https://content.csrc.e1c. nist.gov/News/2016/Post-Quantum-Crypto graphy-Proposed-Requirements

[4] G. Alagic, et al., Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process, National Institute of Standards and Technology (2020). doi: 10.6028/nist.ir.8309

[5] J.H. Cheon, T. Johansson, Post-Quantum Cryptography, in: 13th International Workshop, PQCrypto (2022). doi.org: 10.1007/978-3-031-17234-2.

[6] J.H. Cheon, J.-P. Tillich, Post-Quantum Cryptography, in: 12th International Workshop, PQCrypto, Daejeon, South Korea (2021). doi: 10.1007/978-3-030-81293-5

[7] HQC, URL: https://pqc-hqc.org/ documentation.html

[8] BIKE—Bit Flipping Key Encapsulation. URL: https://bikesuite.org/

[9] Classic McEliece: Talks. https://classic.mceliece.org/talks.html

[10] A.A. Kuznetsov, et al., NIST PQC: Code-based Cryptosystems, TRE. 78 (2019). doi: 10.1615/telecomradeng.v78.i5.50

[11] A. Kuznetsov, et al., Performance Evaluation of the Classic McEliece Key Encapsulation Algorithm, in: 2021 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), (2021) 755–760. doi: 10.1109/idaacs53288. 2021.9660833.

[12] A. Kuznetsov, et al., Code-based cryptosystems from NIST PQC, in: 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT) (2018) 282–287. doi: 10.1109/dessert.2018.8409145.

[13] A. Bessalov, et al., Modeling CSIKE Algorithm on Non-Cyclic Edwards Curves, in: Workshop on Cybersecurity Providing in

Information and Telecommunication Systems, vol. 3288 (2022) 1–10.

[14] A. Bessalov, et al., Computing of Odd Degree Isogenies on Supersingular Twisted Edwards Curves, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems, vol. 2923 (2021) 1–11.

[15] A. Bessalov, V. Sokolov, P. Skladannyi, Modeling of 3- and 5-Isogenies of Supersingular Edwards Curves, in: 2nd International Workshop on Modern Machine Learning Technologies and Data Science, no. I, vol. 2631 (2020) 30–39.

[16] V. Sokolov, P. Skladannyi, H. Hulak, Stability Verification of Self-Organized Wireless Networks with Block Encryption, in: 5th International Workshop on Computer Modeling and Intelligent Systems, vol. 3137 (2022) 227–237.

[17] A. Bessalov, et al., Implementation of the CSIDH Algorithm Model on Supersingular Twisted and Quadratic Edwards Curves, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems, vol. 3187, no. 1 (2022) 302–309.

[18] G. Alagic, et al., Status Report on the First Round of the NIST Post-Quantum Cryptography Standardization Process, National Institute of Standards and Technology (2019). doi: 10.6028/nist.ir.8240

[19] Announcing Request for Nominations for Public-Key Post-Quantum Cryptographic Algorithms, Federal Register (2016). https://www.federalregister.gov/documents/2016/12/20/2016-30615/announcing-request-for-nominations-for-public-key-post-quantum-cryptographic-algorithms

[20] D. Auten, Reconciling Nist's Post-Quantum Cryptography Candidates with Performance Requirements—ProQuest, Southern Illinois University Edwardsville (2020). https://www.proquest.com/openview/9c38158ee73b171bdb6d4f1ff121af64/1?pq-origsite=gscholar&cbl=18750&diss=y

[21] D.J. Bernstein, Introduction to Post-Quantum Cryptography, in: Post-Quantum Cryptography, Springer, Berlin, Heidelberg (2009) 1–14. doi: 10.1007/978-3-540-88702-7_1

[22] I.T.L. Computer Security Division, Round 4 Submissions—Post-Quantum Cryptography, CSRC, NIST (2017). https://csrc.nist.gov/|projects/post-quantum-cryptography/round-4-submissions

[23] R. Overbeck, N. Sendrier, Code-based cryptography, in: Post-Quantum Cryptography, Springer, Berlin, Heidelberg (2009) 95–145. doi: 10.1007/978-3-540-88702-7_4

[24] D.J. Bernstein, J. Buchmann, E. Dahmen, eds., Post-Quantum Cryptography, Springer Berlin Heidelberg (2009). doi: 10.1007/978-3-540-88702-7

[25] M. Bardet, et al., Cryptanalysis of the McEliece Public Key Cryptosystem Based on Polar Codes, in: Post-Quantum Cryptography (2016) 118–143. doi: 10.1007/978-3-319-29360-8_9

[26] Y. Stasev, A. Kuznetsov, Asymmetric Code-Theoretical Schemes Constructed with the Use of Algebraic Geometric Codes, Cybernetics and Systems Analysis 41 (2005) 354–363. doi: 10.1007/s10559-005-0069-9

[27] A. Kuznetsov, et al., Code-based Public-Key Cryptosystems for the Post-Quantum Period, in: 2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PICST) (2017) 125–130. doi: 10.1109/infocommst.2017.8246365.

[28] R.J. McEliece, A Public-Key Cryptosystem Based On Algebraic Coding Theory, Deep Space Network Progress Report 44 (1978) 114–116.

[29] N. Sendrier, Niederreiter Encryption Scheme, in: Encyclopedia of Cryptography and Security (2011) 842–843. doi: 10.1007/978-1-4419-5906-5_385

[30] V Sidelnikov, S. Shestakov, On insecurity of cryptosystems based on generalized Reed-Solomon codes, Discrete Mathematics and Applications. 2 (1992) 439–444. doi: 10.1515/dma.1992.2.4.439

[31] N. T. Courtois, M. Finiasz, N. Sendrier, How to Achieve a McEliece-Based Digital Signature Scheme, in: Advances in Cryptology, ASIACRYPT (2001) 157–174. doi: 10.1007/3-540-45682-1_10