Quantum computer networks and their impact on CyberSecurity: an e-Learning perspective

Valentina MARASCU[1,2], Marius Iulian MIHAILESCU[1], Stefania Loredana NITA[3], Marius ROGOBETE[4], Ciprian RACUCIU[4]

[1] Faculty of Engineering and Computer Science, Scientific Research Center in Mathematics and Computer Science, "Spiru Haret" University, 46G Fabricii, Bucharest, Romania

[2] National Institute for Laser, Plasma and Radiation Physics, 409 Atomistilor Street, Magurele, Romania

[3] Military Technical Academy "Ferdinand I", Bucharest, Romania

[4] Computer Science Faculty, "Titu Maiorescu" University, Bucharest, Romania

valentina.marascu@gmail.com

Abstract: The Quantum Era represents both an opportunity and a challenge for software and hardware fields. The continued developments have produced quantum prototype computers, making them suitable for testing the qubit theory. Thus, by increasing the computational degree, the security part should be considered to produce a secure quantum computational environment. In this line, our paper presents the notions of quantum computing and encryptions, along with dedicated examples, which are helpful for teaching processing. The hands-on example helps the students understand the first steps in the RSA encryption process.

Keywords: quantum computer networks, RSA algorithm, CrypTool environment, education.

1. Introduction

Quantum computers are a new computing device that uses quantum mechanics, a branch of physics that describes the behavior of matter and energy on a small scale. These devices have the potential to revolutionize computing by providing a way to solve particular problems much faster than traditional computers (Djordjevic, 2022) (Koževnikov, 2020). Quantum computer networks are networks of quantum computers connected. These networks enable the collaboration of several quantum computers to address intricate issues that are beyond the capabilities of a single quantum computer (Sutor, 2019), (Van Meter, 2014). In addition to the applications mentioned above, quantum computers possess numerous ancillary use cases. Furthermore, the advent of the quantum era has impacted the 5G and beyond/6G communication systems. Previous generations predominantly depended on symmetric key cryptography for security and privacy. Consequently, in response to the threat posed by quantum computing, mobile broadband standards have opted to transition from symmetric key encryption to PKI-based trust models. Current cryptographic techniques, particularly public key

https://doi.org/10.58503/icvl-v19y202409

encryption and digital signatures, face a significant danger. Symmetric encryption and hash functions remain secure in the post-quantum world. Research on cryptosystems for the post-quantum era is continuing. The pace was sluggish in the early years but has accelerated in recent years. The National Institute of Standards and Technology (NIST) has unveiled a strategy to request, assess, and standardize post-cryptographic algorithms (Chamola et al., 2021), (Suomalainen et al., 2025). One of the most promising quantum computing applications is cryptography and cyber security. Moreover, they possess the capacity to profoundly influence cybersecurity since they can compromise several encryption techniques presently employed to safeguard data (Bertaccini, 2022). These networks are still in their infancy and are presently developed by researchers and technology companies worldwide. The main advantage of quantum computer networks is their ability to solve particular problems much faster than classical computers. Quantum computers are particularly suited to solving optimization, machine learning, and cryptography problems. Encryption is converting data into a code that unauthorized parties cannot read (Mihailescu & Nita, 2023). Encryption protects sensitive information such as financial transactions, medical records, and government secrets. Currently, encryption is based on mathematical problems that are difficult to solve, even for traditional computers. Quantum computers have the ability to solve some problems at a significantly quicker rate than traditional computers, hence rendering many existing encryption schemes ineffective (Mihailescu & Nita, 2021). Quantum computers can render the RSA algorithm ineffective, which is a commonly employed encryption method for safeguarding sensitive data. The RSA algorithm relies on the computational challenge of decomposing big integers into their prime components. Conventional computers are capable of solving this issue; however, when dealing with big numbers, the computational time required is significantly increased (Nita & Mihailescu, 2022). Quantum computers have the capability to solve this issue at a much-accelerated rate, enabling them to break RSA encryption. Quantum computers have the capability to decrypt the elliptic curve cryptography (ECC) technique. This algorithm ensures the security of data transmitted over the Internet, including the establishment of secure connections between websites and users (Grasselli, 2021). ECC encryption is based on the computational complexity of specific mathematical equations. However, the advent of quantum computers has introduced a significant challenge since they are capable of solving these equations at a considerably quicker rate compared to ordinary computers. Consequently, quantum computers have the potential to compromise ECC encryption (Bassoli et al., 2021), (Wolf, 2021). Based on the above information, e-learning computational quantum techniques and applying them in the real-world, will produce many challenges for both educators and students. The difference between bit and qubit and how the quantum era will influence the computation ability will highlight new teaching methods in the frame of e-learning. Our paper presents a theoretical-practical method in which the student has both theory and hands-on experience via the CrypTool environment. The great advantage of this is that students can access the information at any time. The

e-Learning platform is designed to offer information, answers, and hands-on demonstrations.

2. The RSA (Rivest-Shamir-Adleman) algorithm

The RSA algorithm, also known as Rivest-Shamir-Adleman, is a commonly utilized cryptographic technique for ensuring safe transmission of data over the Internet through the use of public-key encryption. The cryptographic method was created in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman. Since then, it has gained immense popularity and is now widely utilized worldwide. The system operates based on the notion of utilizing two distinct keys: a public key and a private key. The public key is employed for data encryption, while the private key is utilized for data decryption. The public key can be freely disseminated, but the private key must be securely guarded. The security of RSA relies on the arduousness of decomposing huge composite numbers into their prime elements. The algorithm derives public and private keys by choosing two large prime integers, multiplying them, and utilizing the product as the modulus for both the public and private keys. The encryption strength is determined by the length of the key, with longer keys typically providing greater security. In order to employ RSA encryption, the sender utilizes the public key of the receiver to encrypt the message. The communication can only be decrypted by the receiver who possesses the associated private key. This guarantees that the message can only be viewed by the designated recipient, even if a third party intercepts it (Grimes, 2019), (Aumasson & Green, 2017).

2.1 The mathematical approach of the RSA algorithm

The RSA algorithm is a type of cryptographic technique that uses asymmetric encryption. Asymmetric cryptography utilizes a pair of distinct keys, namely the public key and the private key. The public key is distributed to all individuals, whereas the private key is maintained in secrecy, as its name suggests. The following situation can show an instance of asymmetric cryptography:

- A client (C), such as a browser, transmits its public key to the server (S) and makes a request for specific data;
- The server (S) employs the client's public key to encrypt the data and subsequently transmits the encrypted data;
- The data is received by the client and subsequently decrypted.

Due to its asymmetry, only the browser can decode the data, even if a third party possesses the public key of the browser. The concept of RSA is founded on the inherent complexity of factoring big integers.

The public key comprises two integers, one being the result of multiplying two huge prime numbers. The private key is formed from the identical pair of

prime numbers. Therefore, if an individual is able to factorize the huge number, the private key becomes compromised. Hence, the level of encryption is directly proportional to the size of the key, and by increasing the key size by two or three times, the encryption strength grows exponentially. RSA keys commonly have lengths of either 1024 or 2048 bits. Nevertheless, analysts anticipate that 1024-bit keys may soon become vulnerable to decryption. However, up to this point, it seems to be an unattainable undertaking.

2.1.1 Public key generation

The process of generating the public key consists of the following steps:

- Choose two primary integers. Assume that $A = 41$ and $B = 47$.
- The initial component of the public key is $c = A \cdot B = 1927$.
- Additionally, we require a diminutive exponent, such as f, with the condition that f must be an integer. It is not necessary for it to be a factor of $\Delta(c)$.

$$1 < f < \Delta(c) \quad (1)$$

2.1.2 Private key generation

The process of generating the private key entails the following steps:
To find $\Delta(c)$, we must compute the value of $(A-1)(B-1)$.
The private key, g, can be calculated as follows: $g = h \cdot \Delta(c) + 1$.

2.2 RSA using CrypTool environment

CryptoTool is a free, open-source software tool with cryptographic and cryptanalytic functionality. It is designed to help people learn and understand cryptographic concepts by providing an easy-to-use interface for experimenting with cryptographic algorithms and analyzing their security. The software includes a variety of functions, such as encrypting and decrypting messages, generating cryptographic keys, and analyzing cryptographic protocols. It supports many cryptographic algorithms, including symmetric ciphers, asymmetric ciphers, hash functions, and digital signatures. It is commonly used by students, educators, and researchers in cryptography and information security. It is available for download on various platforms, including Windows, Linux, and macOS. CrypTool has four versions.

CrypTool versions are:

- CrypTool-Online (*CTO* for short) offers applications for testing, learning, and discovering ancient and modern cryptography. It can be accessed at: https://www.cryptool.org/en/cto/;
- CrypTool 1 (CT1) is the inaugural iteration of CrypTool, which was created in

1998 utilizing the C++ programming language. This is a no-cost software for cryptography and cryptanalysis that is compatible with Windows operating systems. CT1 is accessible in six languages and is the most used e-learning package. This can be accessed at: https://www.cryptool.org/en/ct1/ ;

- CrypTool 2 (CT2) is a contemporary educational software created with Microsoft's .NET technology and the C# programming language. It enables the observation of cryptography and cryptanalysis. The subject matter encompasses the study of encryption and cryptanalysis of ciphers, including its fundamental components, as well as the entirety of contemporary cryptography. This can be accessed at: https://www.cryptool.org/en/ct2/ ;
- JCrypTool (JCT) is a user-friendly program that enables students, professors, developers, and cryptography enthusiasts to implement and examine cryptographic methods. The JCT platform revolutionizes e-learning by promoting user engagement in cryptography, algorithm use, cryptographic plugin development, and expansion of the JCrypTool platform into novel domains. JCrypTool achieved its stable version 1.0.0 in November 2020. The majority of its functionality is implemented through more than 100 distinct plugins. This can be seen at: https://www.cryptool.org/en/jct/.

2.2.1 RSA Algorithm with *CryptTool 1 (CT1)* environment

The steps to perform encryption and decryption operations with *RSA* using *CT1* are as follows:

1. We open *CT1*. Either from the Desktop or from the Start menu of the Windows operating system;

2. The application opens. We familiarize ourselves with the platform environment and the desktop application;

3. From the individual menu (see figure 1). for the procedures, select rsa cryptosystem – RSA demonstration...

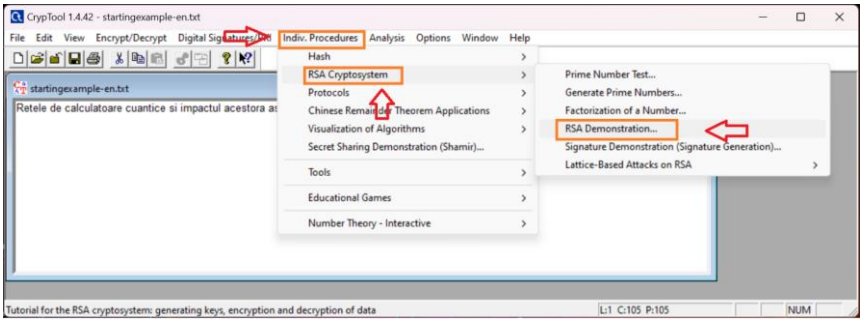

Figure 1. Platform window of the CT1 software

4. Review the RSA demonstration window. We familiarize ourselves with the elements and data that must be generated automatically or filled in manually, making a parallel with the mathematical apparatus of the RSA algorithm presented above.

5. We click on the Generate prime numbers button, and the window opens, where we can generate two different prime numbers. We will select the *Miller-Rabin Test* to generate and test the primality of numbers. Afterward, we will leave the upper and lower bounds for the two numbers (*p and q*) as they are and click Generate prime numbers, and the output should be similar to Figure 2. The next step is to press the Apply button received.

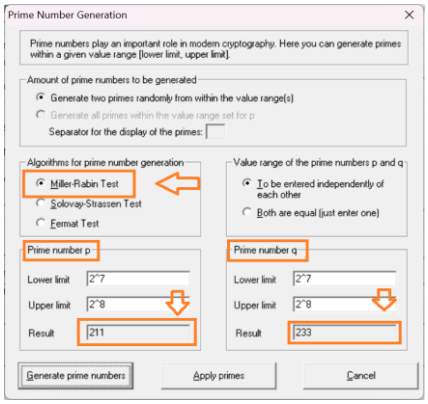

Figure 2. Generation of prime numbers used to encrypt the message

6. We notice the changes made in the RSA Demonstration window. We analyze the obtained RSA parameters and correlate these values with the abovementioned mathematical concepts.

7. In the Enter the message for encryption (see Figure 3) or decryption either as text or as hex dump field, enter the text/word we want to encrypt (without the quotes): “Quantum Computer Networks and Their Impact on CyberSecurity by V. Marascu, M. I. Mihailescu, S. L. Nita, M. Rogobete and C. Racuciu” and click on Encrypt button. We observe the changes made:

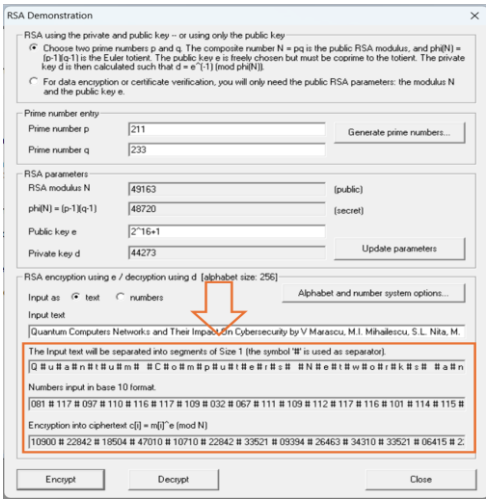

Figure 3. Generation of prime numbers used to encrypt the message.

8. For decryption, we will need to know the values of *p* and *q* and the encrypted version of the message displayed in the field encryption into ciphertext *c[i]=m[i]^e (mode n)*. It must be mentioned that the encryption process is done character by character. For the message chosen above (in step 7) for encryption, the encrypted version is: "Quantum Computers Networks and Their Impact on Cybersecurity by V. Marascu, M. I. Mihailescu, S. L. Nita, M. Rogobete and C. Racuciu".

The results of the encrypted text via:

a) the Miller-Rabin Test:

10900 # 22842 # 18504 # 47010 # 10710 # 22842 # 33521 # 09394 # 26463 # 34310 # 33521 # 06415 # 22842 # 10710 # 07428 # 08293 # 06205 # 09394 # 29564 # 07428 # 10710 # 00095 # 34310 # 08293 # 30861 # 06205 # 09394 # 18504 # 47010 # 42283 # 09394 # 00500 # 23366 # 07428 # 20714 # 08293 # 09394 # 37508 # 33521 # 06415 # 18504 # 35574 # 10710 # 09394 # 39271 # 47010 # 09394 # 26463 # 02206 # 24394 # 07428 # 08293 # 06205 # 07428 # 35574 # 22842 # 08293 # 20714 # 10710 # 02206 # 09394 # 24394 # 02206 # 09394 # 14261 # 09394 # 16226 # 18504 # 08293 # 18504 # 06205 # 35574 # 22842 # 47242 # 09394 # 16226 # 04897 # 37508 # 04897 # 09394 # 16226 # 20714 # 23366 # 18504 # 20714 # 41521 # 07428 # 06205 # 35574 # 22842 # 47242 # 09394 # 00559 # 04897 # 16252 # 04897 # 09394 # 29564 # 20714 # 10710 # 18504 # 47242 # 09394 # 16226 # 04897 # 09394 # 25674 # 34310 # 12284 # 34310 # 24394 # 07428 # 10710 # 07428 # 09394 # 18504 # 47010 # 42283 # 09394 # 26463 # 04897 # 09394 # 25674 # 18504 # 35574 # 22842 # 35574 # 20714 # 22842 # 09394

b) the Solovay-Strassen Test:

45834 # 23248 # 45045 # 48257 # 53474 # 23248 # 26521 # 41646 # 48332 # 48449 # 26521 # 03060 # 23248 # 53474 # 07539 # 51764 # 45972 # 41646 # 44412 # 07539 # 53474 # 49310 # 48449 # 51764 # 43808 # 45972 # 41646 # 45045 # 48257 # 14081 # 41646 # 27305 # 51552 # 07539 # 37343 # 51764 # 41646 # 49873 # 26521 # 03060 # 45045 # 22028 # 53474 # 41646 # 21701 # 48257 # 41646 # 48332 # 39074 # 33764 # 07539 # 51764 # 45972 # 07539 # 22028 # 23248 # 51764 # 37343 # 53474 # 39074 # 41646 # 33764 # 39074 # 41646 # 28035 # 41646 # 48206 # 45045 # 51764 # 45045 # 45972 # 22028 # 23248 # 21969 # 41646 # 48206 # 32659 # 49873 # 32659 # 41646 # 48206 # 37343 # 51552 # 45045 # 37343 # 38993 # 07539 # 45972 # 22028 # 23248 # 21969 # 41646 # 41864 # 32659 # 02070 # 32659 # 41646 # 44412 # 37343 # 53474 # 45045 # 21969 # 41646 # 48206 # 32659 # 41646 # 41889 # 48449 # 00089 # 48449 # 33764 # 07539 # 53474 # 07539 # 41646 # 45045 # 48257 # 14081 # 41646 # 48332 # 32659 # 41646 # 41889 # 45045 # 22028 # 23248 # 22028 # 37343 # 23248 # 41646

b) the Fermat Test:

10575 # 04473 # 15333 # 06712 # 02704 # 04473 # 05718 # 14301 # 04934 # 05439 # 05718 # 02857 # 04473 # 02704 # 18091 # 13208 # 12444 # 14301 # 05703 # 18091 # 02704 # 01760 # 05439 # 13208 # 01128 # 12444 # 14301 # 15333 # 06712 # 05179 # 14301 # 11758 # 02004 # 18091 # 05469 # 13208 # 14301 # 14929 # 05718 # 02857 # 15333 # 10631 # 02704 # 14301 # 02886 # 06712 # 14301 # 04934 # 20184 # 05664 # 18091 # 13208 # 12444 # 18091 # 10631 # 04473 # 13208 # 05469 # 02704 # 20184 # 14301 # 05664 # 20184 # 14301 # 06028 # 14301 # 20701 # 15333 # 13208 # 15333 # 12444 # 10631 # 04473 # 13007 # 14301 # 20701 # 20301 # 14929 # 20301 # 14301 # 20701 # 05469 # 02004 # 15333 # 05469 # 05372 # 18091 # 12444 # 10631 # 04473 # 13007 # 14301 # 07087 # 20301 # 10103 # 20301 # 14301 # 05703 # 05469 # 02704 # 15333 # 13007 # 14301 # 20701 # 20301 # 14301 # 08923 # 05439 # 03688 # 05439 # 05664 # 18091 # 02704 # 18091 # 14301 # 15333 # 06712 # 05179 # 14301 # 04934 # 20301 # 14301 # 08923 # 15333 # 10631 # 04473 # 10631 # 05469 # 04473 # 14301

3. Shor's algorithm (the quantum approach of the RSA algorithm)

Quantum RSA, or Shor's algorithm, is a computational method in quantum computing that may effectively decompose huge numbers into their prime components. Mathematician Peter Shor invented this method in 1994, which is widely regarded as one of the most crucial quantum algorithms. The RSA algorithm is extensively employed in contemporary cryptography for the purpose of encrypting and decrypting data. The process relies on decomposing big numbers into their prime components, a task that is arduous and time-consuming for conventional computers. However, using Shor's algorithm, quantum computers can accomplish this task much faster. This means that the development of quantum computers threatens the security of RSA encryption. If a large-scale quantum computer is built, it could easily break RSA encryption and access sensitive data such as financial transactions and personal information. To address this problem, researchers are developing new quantum-resistant encryption methods that are not vulnerable to quantum computer attacks. The newly developed techniques, such as network-based encryption and code-based cryptography, have been specifically intended to provide strong security against potential assaults from both conventional and quantum computers. The Shor algorithm, also known as the Shor-2 algorithm, is a fast integer factorization algorithm. It was developed by mathematician and computer scientist Daniel Shor in 1981 and improved the original Shor algorithm. It is based on the quadratic stack algorithm and uses the same basic ideas. It works by finding a set of smooth numbers that can be used to construct a unity between two quadratic residuals. This congruence can then be used to factor the original integer. Shor's algorithm is particularly efficient for factoring integers with small factors and has been used to factor several RSA challenge numbers. However, it is not as widely used as other factorization

algorithms, such as the general number field stack (GNFS), currently the fastest known algorithm for factoring large integers. Overall, Shor's algorithm represents a significant contribution to integer factorization and helped pave the way for further developments in this area of research.

The problem we are trying to solve is, given an odd composite number *N*, to factor *N*. Shor's algorithm consists of two parts:

- The factorization problem may be reduced to the order-finding problem using classical methods.
- An algorithm utilizing quantum computing to solve the order search issue.

The reduction in Shor's factorization algorithm is analogous to other factorization algorithms, such as the quadratic stack.

3.1 Classical Procedure

1. Take a pseudo-random number $a < N$.

2. Calculate *GCD (a, N).* This can be done using Euclid's algorithm.

3. If $GCD\ (a, N) \neq 1$, then it is a factor of *N*, which provides a solution to the problem.

4. Otherwise, use the period search subroutine (below) to find *r*, the period of the function, that is, the smallest *r* integer.

5. If *r* is odd, go back to step 1.

6. If $ar / 2 \equiv -1\ (mod\ N)$, return to step 1.

7. The *N* factors are $GCD\ (ar / 2 \pm 1.\ N)$, which solves the problem.

3.2 The quantum part: the periodic probe subroutine

This subroutine consists of the following steps:

1. Start with the input and output registers of each log *2N* qubits and initialize them to where *x* goes from *0* to $N - 1$, where *x* goes from *0* to *N-1*

$$N^{-\frac{1}{2}} \sum_{x} |x\rangle\, |0\rangle \tag{2}$$

2. Construct *f(x)* as a quantum function and apply it to the previous state to obtain

$$N^{-\frac{1}{2}} \sum_{x} |x\rangle\, |f(x)\rangle \tag{3}$$

3. Apply the quantum Fourier transform to the input register. The quantum Fourier transform on *N* points is defined by:

$$UFQT\,|x\rangle = N^{-\frac{1}{2}} \sum_{y} e^{\frac{2\pi i x y}{N}} |y\rangle \tag{4}$$

Which provides the following state:

$$N^{-1} \sum_{x} \sum_{y} e^{\frac{2\pi i x y}{N}} |y\rangle |f(x)\rangle \tag{5}$$

4. Take a measurement. This gives a specific value, *y*, in the input and output registers. Since *f* is periodic, the probability of measuring a given *y* is given by

$$\left| N^{-1} \sum_{x:f(x)=f(x_0)} e^{\frac{2\pi i x y}{N}} \right|^2 = \left| N^{-1} \sum_{b} e^{\frac{2\pi i (x_0 + r b) y}{N}} \right|^2 \tag{6}$$

5. The calculation shows that this probability is higher when *yr / N* is close to an integer.

6. Check that *f(x) = f(x + r').* If so, it's over.

7. Otherwise, get more candidates for *r* using close values of *y* or multiples of r' . If another candidate goes, it's over.

8. Otherwise, return to step 1 of the routine.

4. Quantum cryptography based on elliptic curves

Quantum elliptic curve cryptography (QECC) is a kind of public key cryptography that relies on the mathematical features of elliptic curves. Quantum Error Correction Codes (QECC) are extensively employed in contemporary cryptography to ensure the security of Internet communications, including the establishment of secure connections between websites and users. Elliptic curve cryptography (ECC) is a cryptographic technique that utilizes the features of elliptic curves for public key encryption. ECC uses the difficulty of solving some mathematical issues to encrypt and decrypt data. Like RSA, ECC is vulnerable to quantum computer attacks. Quantum computers can solve math problems used in ECC much faster than classical computers. This means that the development of quantum computers threatens the security of ECC. To solve this problem, researchers are developing new quantum-resistant cryptography methods that are not vulnerable to quantum computer attacks. One approach is to use isogeny-based cryptography, which is a form of cryptography based on the properties of elliptic curves but is resistant to quantum computer attacks. Isogeny-based encryption utilizes the mathematical features of isogenies, which are mappings across elliptic

curves that maintain certain algebraic properties. Isogeny-based encryption is specifically developed to provide security against both conventional and quantum computing threats. Quantum elliptic curve cryptography (QECC) is a kind of public key cryptography that is susceptible to exploitation by quantum computers. Current investigations in quantum-resistant cryptography strive to create novel techniques, such as isogeny-based encryption, that will effectively withstand assaults from both classical and quantum computers.

The shift to quantum cryptography poses both an opportunity and a difficulty for contemporary computer science students. Several considerations are relevant from their viewpoint:

- Students will be excited about the shift to quantum cryptography, being seen as a huge evolution, such as the transition from mechanical to digital systems;
- Concepts like qubits, superpositions, and entanglement are hard to understand by the students, if they are using the classical concepts.
- Companies such as IBM and Google invest in quantum technologies, this being an important argument for the students in the frame of quantum technologies learning.

5. Conclusions

The present manuscript represents an educational approach to quantum computing and encryption processes, where the students have the opportunity to understand the concepts, along with dedicated descriptions. The theoretical approach is combined with hands-on examples, making them suitable for the educational process. Students have the possibility to make connections between the classical approach and the quantum approach. Also, by inserting original print screens of the used software for the encryption process, the students have the chance to learn step-by-step instructions for individual learning. Further works will include more complex encryption methods where the students can observe in real time how the qubit is used during the computational process.

REFERENCES

Aumasson, J.-P. & Green, M. D. (2017) *Serious Cryptography: A Practical Introduction to Modern Encryption*. No Starch Press.

Bassoli, R., et al. (2021) *Quantum Communication Networks*. Springer.

Bertaccini, M. (2022) *Cryptography Algorithms: A Guide to Algorithms in Blockchain, Quantum Cryptography, Zero-Knowledge Protocols, and Homomorphic Encryption*. Packt.

Chamola, V. et al. (2021) Information security in the post quantum era for 5G and beyond networks: Threats to existing cryptography, and post-quantum cryptography. *Computer Communications*. 176, 99–118. doi:10.1016/j.comcom.2021.05.019 .

Djordjevic, I. (2022) *Quantum Communication, Quantum Networks, and Quantum Sensing*. Academic Press.

Grasselli, F. (2021) *Quantum Cryptography: From Key Distribution to Conference Key Agreement*. Springer.

Grimes, R. A. (2019) *Cryptography Apocalypse: Preparing for the Day When Quantum Computing Breaks Today's Crypto*. John Wiley & Sons Inc.

Koževnikov, A. B. (2020) *The Copenhagen Network: The Birth of Quantum Mechanics from a Postdoctoral Perspective*. Springer, 2020.

Mihailescu, M. I. & Nita, S. L. (2021) Pro Cryptography and Cryptanalysis with C++20: Creating and Programming Advanced Algorithms. *Apress*.

Mihailescu, M. I. & Nita, S. L. (2023) *Pro Cryptography and Cryptanalysis with C++23: Creating and Programming Advanced Algorithms*. Apress.

Nita, S. L. & Mihailescu, M. I. (2022) Cryptography and Cryptanalysis in Java: Creating and Programming Advanced Algorithms with Java SE 17 LTS and Jakarta EE 10. Apress.

Suomalainen, J., Ahmad, I., Shajan, A. & Savunen, T. (2025) Cybersecurity for tactical 6G networks: threats, architecture, and intelligence. *Future Generation Computer Systems.* 162, 107500. doi:10.1016/j.future.2024.107500.

Sutor, R. S. (2019) *Dancing with Qubits: How Quantum Computing Works and How It May Change the World*. Packt, 2019.

Van Meter, R. (2014) *Quantum Networking*. ISTE. Wiley, 2014.

Wolf, R. (2021) *Quantum Key Distribution: An Introduction with Exercises*. Springer.