Communication Security in Computer Networks: Encryption Methods and Performance Evaluation

Constantin-Alin COPACI, Dorina-Luminița COPACI
Faculty of Electronics, Telecommunications and Information Technology,
National University of Science and Technology POLITEHNICA Bucharest, Romania
constantin.copaci@stud.etti.upb.ro, dorina.copaci@upb.ro

Abstract

With the increasing complexity and interconnectivity of networks, numerous cyber threats have emerged, including data interception, packet manipulation, man-in-the-middle attacks, spoofing, and Denial-of-Service (DoS) attacks. In response to these risks, data encryption stands out as one of the most effective strategies for safeguarding the confidentiality and integrity of information transmitted over networks. This study aims to analyze and compare several encryption methods employed in securing communications within computer networks, using simulations conducted in the MATLAB environment. Data encryption scenarios will be implemented, and the performance of the algorithms will be assessed based on metrics such as execution time, key size, and the impact on communication latency.

Index terms: cryptography, cyber security, encryption, latency

1. Introduction

Modern cryptography enables the transformation of data into a format that is unreadable to unauthorized third parties, with algorithms such as AES (Advanced Encryption Standard) and RSA widely implemented across IT infrastructures. Selecting an appropriate encryption algorithm, as well as understanding the trade-offs between security levels and system performance, are critical considerations in designing a reliable and secure communication system [1].

This study is organized into four chapters. Chapter 2 introduces the fundamental concepts of computer networks and their security. Chapter 3 provides a detailed overview of encryption methods employed in communications. Chapter 4 focuses on MATLAB simulations and the performance evaluation of the algorithms. The work concludes with key findings and prospective directions for future research.

2. Fundamentals of computer networks

Computer networks enable the exchange of data between devices through standardized protocols and are essential for modern communications. The TCP/IP model, which predominates in contemporary infrastructures, manages data transmission via the network layer (IP) and transport layer (TCP/UDP), supporting higher-level protocols such as HTTP and FTP [2].

The main types of networks include local area networks (LANs), wide area networks (WANs), and wireless networks, each with characteristics that influence communication security. Typical threats in these environments range from passive interception (sniffing) to active attacks such as man-in-the-middle or denial-of-service (DoS) attacks [3].

To safeguard the integrity and confidentiality of transmitted data, encryption is a critical component of network security. In the context of this study, MATLAB simulations [4] will model data encryption within network protocols, evaluating the impact of various algorithms on communication performance, including latency and resource consumption.

This theoretical foundation is essential for understanding the rationale behind implementing and testing cryptographic mechanisms in networks, as explored in the subsequent chapters.

3. Encryption methods employed in computer networks

The security of communications in computer networks fundamentally relies on cryptography, which protects data by transforming it into forms unreadable to unauthorized parties. Encryption can be broadly categorized into two main types: symmetric encryption and asymmetric encryption, each serving specific roles and applications within security protocols [1], [5], [6].

Symmetric encryption

Symmetric encryption uses the same secret key for both encrypting and decrypting data. This method is computationally efficient and fast, making it commonly employed to secure large volumes of data transmitted over a network.

The Advanced Encryption Standard (AES) is the current benchmark for symmetric encryption, widely adopted due to its robust security and high performance. AES operates on 128-bit blocks and employs key sizes of 128, 192, or 256 bits for data encryption and decryption. The algorithm involves multiple rounds of substitution, permutation, and mathematical transformations, providing strong resistance against brute-force attacks and statistical analysis.

In networking contexts, AES is extensively used for encrypting traffic at the level of TLS protocols, Virtual Private Networks (VPNs), and for securing stored data.

Asymmetric encryption

Asymmetric encryption relies on a pair of keys: a public key, which is accessible to anyone and used for data encryption, and a private key, kept confidential and used for decryption. While this method is generally slower than symmetric encryption, it is indispensable for secure key exchange and authentication.

The RSA (Rivest-Shamir-Adleman) algorithm [1] is one of the most widely adopted asymmetric encryption techniques, based on the computational difficulty of factoring large prime numbers. The generation of the key pair involves selecting two large prime numbers, with security largely dependent on their length, typically 2048 bits or higher.

RSA is employed in protocols such as SSL/TLS for secure symmetric key exchange, authentication, and digital signatures.

In most practical scenarios, symmetric and asymmetric encryption are combined to leverage the strengths of both methods. This hybrid approach enhances both security and system performance.

In addition to classical algorithms, modern networks can also employ other cryptographic techniques to secure communications:

- Cryptographic hashing (e.g., SHA-256) for ensuring data integrity and verifying authenticity.
- Cryptographic Protocols such as TLS/SSL and IPsec, which integrate encryption and authentication methods.
- Elliptic curve cryptography (ECC), providing high levels of security with shorter keys and improved performance.

For the evaluation of network security in this study, the AES and RSA algorithms will be employed in simulations. MATLAB [4] is used as the software platform to assess performance in

computer networks. The evaluation will focus on measuring encryption and decryption times, the impact on transmission latency, and resource consumption, providing a practical perspective on the efficiency of these methods in real-world environments.

4. Implementation and simulation of encryption methods in MATLAB

To practically evaluate the performance and efficiency of the AES and RSA encryption methods, this study uses the MATLAB environment, known for its flexibility and capability to simulate complex communication systems [4]. Implementing the AES and RSA algorithms in MATLAB allows for the analysis of operational parameters under controlled conditions, providing relevant data on latency, resource consumption, and impact on network transmission.

For the conducted simulations, the following MATLAB toolboxes and functions were used:

- Communications Toolbox: for generating and manipulating data packets, simulating channels, and evaluating communication performance.
- Custom cryptographic functions: since MATLAB does not natively provide all functions for AES and RSA encryption, custom scripts and functions were implemented or third-party integrated libraries were used.
- Standard MATLAB functions for measuring execution time (tic and toc) and data management.

4.1. Implementation of the AES algorithm

The AES algorithm [1], [7] is implemented according to the standard specifications, with the following steps:

- Processing the input data into 128-bit blocks.
- Applying rounds of substitution, permutation, and mathematical transformations.
- Using 128-bit keys for encryption and decryption.
- Validating functionality by encrypting and decrypting test messages.

For each encrypted message, the processing time is measured, and the results are stored for analysis.

To implement the algorithm, we first define the text we want to encrypt:

plaintext = 'Criptare AES in retelele de calculatoare';

Each character is converted into its ASCII code, that is, into a number between 0 and 255 (one byte). AES operates on bytes, not directly on characters.

data = uint8(plaintext);

The AES key must be exactly 16 bytes for AES-128. A fixed, complex key with varied values was chosen (in hex: 3A 7F E1 4C 9B 20 D3 A8 5E 11 B2 6C 99 47 FF 02).

```
% Hex: 3A 7F E1 4C 9B 20 D3 A8 5E 11 B2 6C 99 47 FF 02
key = uint8([58, 127, 225, 76, 155, 32, 211, 168, 94, 17, 178, 108, 153, 71, 255, 2]);
```

AES encrypts 16-byte blocks. If the text length is not a multiple of 16, padding is added. If the length is already a multiple of 16, a full padding block (16 bytes) is added, according to the PKCS#7 standard. The padding consists of bytes all equal to the number of bytes added.

For encryption, a cipher object is created using: AES, ECB (Electronic Codebook) mode, and PKCS5 padding (equivalent to PKCS7 for 16-byte blocks). The cipher is initialized for encryption with the key. The data with padding is then encrypted, resulting in the ciphertext.

```
% Criptare (AES/ECB/PKCS5Padding)
cipher = Cipher.getInstance('AES/ECB/PKCS5Padding');
cipher.init(Cipher.ENCRYPT_MODE, secretKey);
ciphertext = cipher.doFinal(paddedData);
```

Decryption involves specifying the object to be decrypted and issuing the decryption command.

```
% Decriptare
cipher.init(Cipher.DECRYPT_MODE, secretKey);
decryptedPadded = cipher.doFinal(ciphertext);
```

AES helps secure networks by maintaining confidentiality and restricting unauthorized access. It is a key component in protecting communications, but it must be used correctly (with secure modes, strong keys, proper key management, etc.).

At the same time, we measure the encryption and decryption times, the impact on transmission latency, and resource consumption, providing a practical perspective on the efficiency of these methods in real-world environments (Fig. 1, Fig. 2, Fig. 3).

```
===== Rezultate criptare AES =====
Text original: Criptare AES in retelele de calculatoare
Text decriptat: Criptare AES in retelele de calculatoare
Cheie (hex): 3A 7F E1 4C 9B 20 D3 A8 5E 11 B2 6C 99 47 FF 02
Text criptat (hex): -1.400000e+01 -2.000000e+01 42 -9.000000e+01 07 -8.700000e+01 3D 4F 66 -8.900000e
Dimensiune text criptat: 64 bytes

===== Timpi măsurați =====
Timp criptare:    0.000260 sec
Timp decriptare:  0.000340 sec
Timp transmisie fără criptare: 0.029201 sec
Timp transmisie cu criptare:   0.016862 sec
Diferență de latență (simulată): -0.012339 sec
```

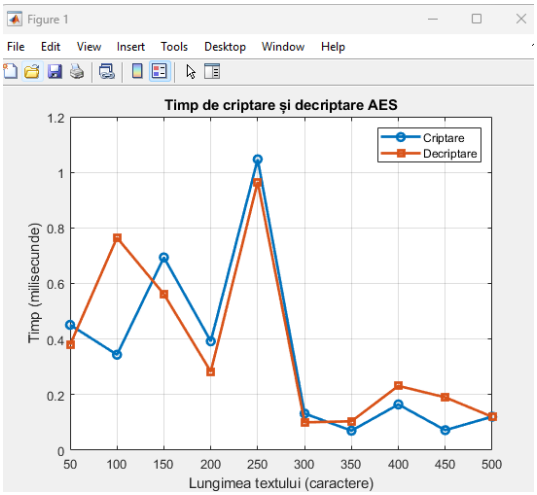

Fig. 1. Encryption time and decryption time using the AES algorithm

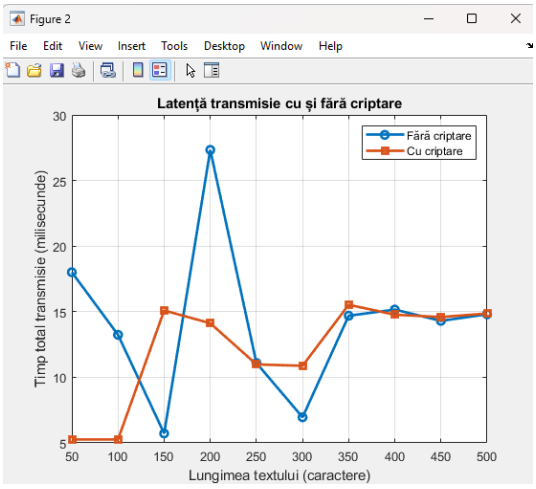

Fig. 2. Transmission latency with and without AES-ECB encryption

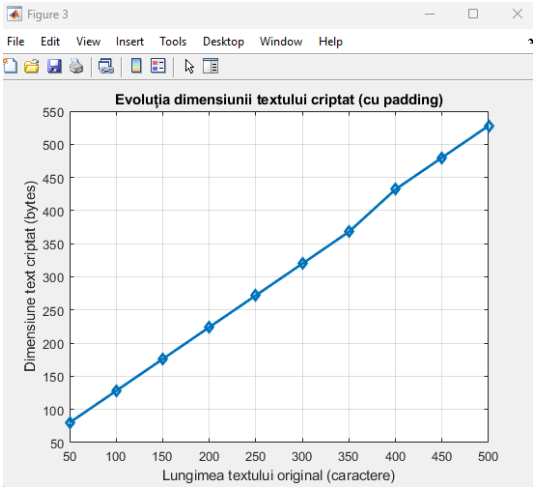

Fig. 3. Evolution of the ciphertext size using AES-ECB

Figures 4 and 5 show the encryption and decryption times, and the ciphertext size, respectively, using AES CBC (Cipher Block Chaining) encryption:

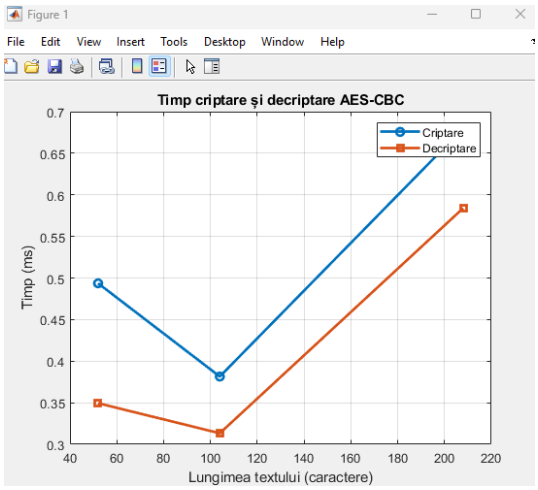

Fig. 4. AES-CBC encryption and decryption times

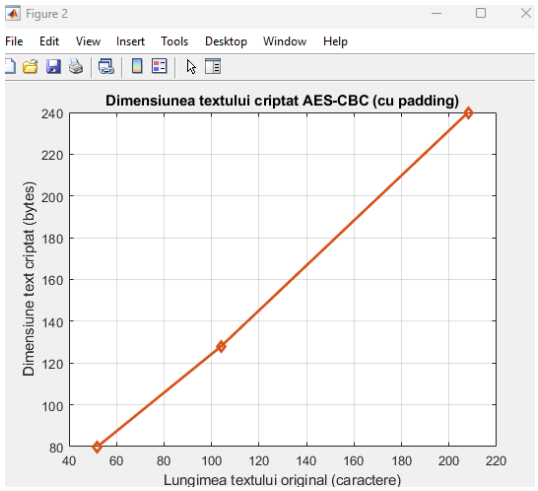

Fig. 5. AES-CBC ciphertext size

By comparing the AES ECB (Electronic Codebook) and AES CBC (Cipher Block Chaining) algorithms, it can be observed that CBC is more secure than ECB, since ECB encrypts each block identically, making patterns in the data detectable (Fig. 6), whereas CBC uses an Initialization Vector (IV) and links each block to the previous one, eliminating repetitions in the ciphertext (Fig. 7).

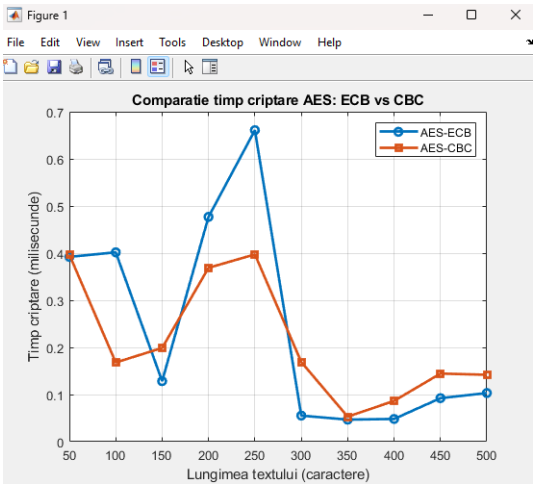

Fig. 6. Encryption time comparison: AES ECB vs. CBC

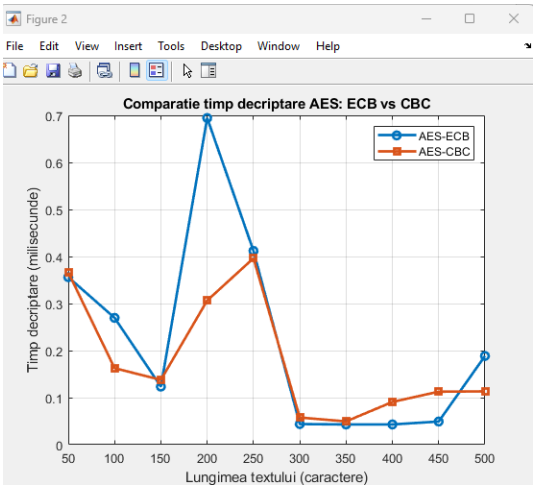

Fig. 7. Decryption time comparison: AES ECB vs. CBC

4.2. Implementation of the RSA algorithm

RSA is an asymmetric encryption algorithm: it uses two keys, a public key (for encryption) and a private key (for decryption), and relies on the difficulty of factoring large numbers.

The implementation of the RSA algorithm involves the following steps:

1. Choose two large prime numbers, p and q;
2. Compute $n = p \times q$;
3. Calculate $\Phi(n) = (p-1) \times (q-1)$;
4. Choose the public exponent e $(co-prime\ with\ \Phi(n)$;
5. Compute the private key d such that $e \cdot d \equiv 1 mod \Phi(n)$;
6. Encrypt $C = M^e mod\ n$;
7. Decrypt $M = C^d mod\ n$.

Fig. 8 shows the implementation of the RSA algorithm for message encryption and decryption.

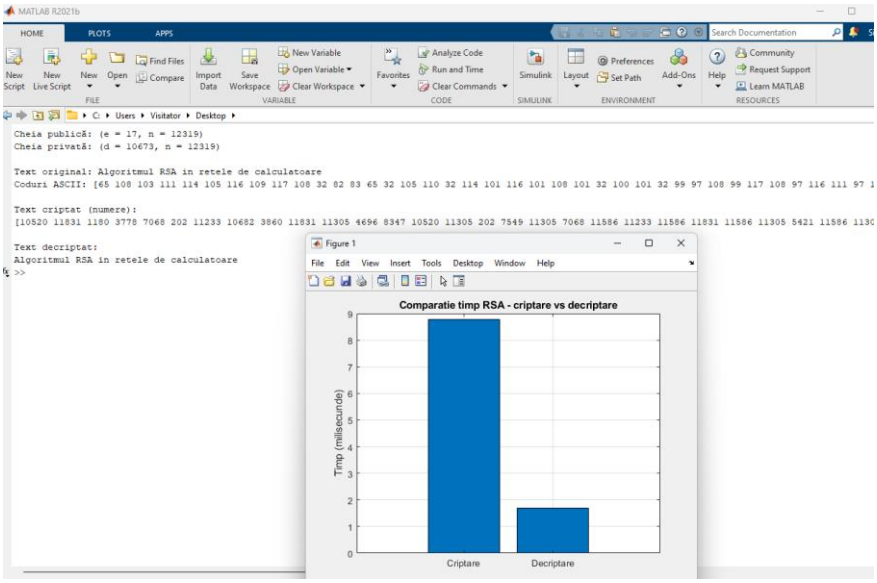

Fig. 8. RSA encryption/decryption time comparison

The RSA algorithm is one of the important cryptographic technologies used in networks and the Internet, as it ensures confidentiality - data encrypted with the public key can only be read with the private key; authentication - the message can be signed with the private key; and integrity - any modification of digitally signed data can be detected.

4.3. Comparison between AES and RSA algorithms in a network with packet loss

AES and RSA are two fundamental algorithms for data encryption. AES is a symmetric algorithm, using a single key for both encryption and decryption, while RSA is an asymmetric algorithm, using a pair of keys (public and private). In computer networks, encryption security and performance are essential for protecting data. Comparing these two algorithms is an important method for evaluating the security of communication networks (Fig. 9).

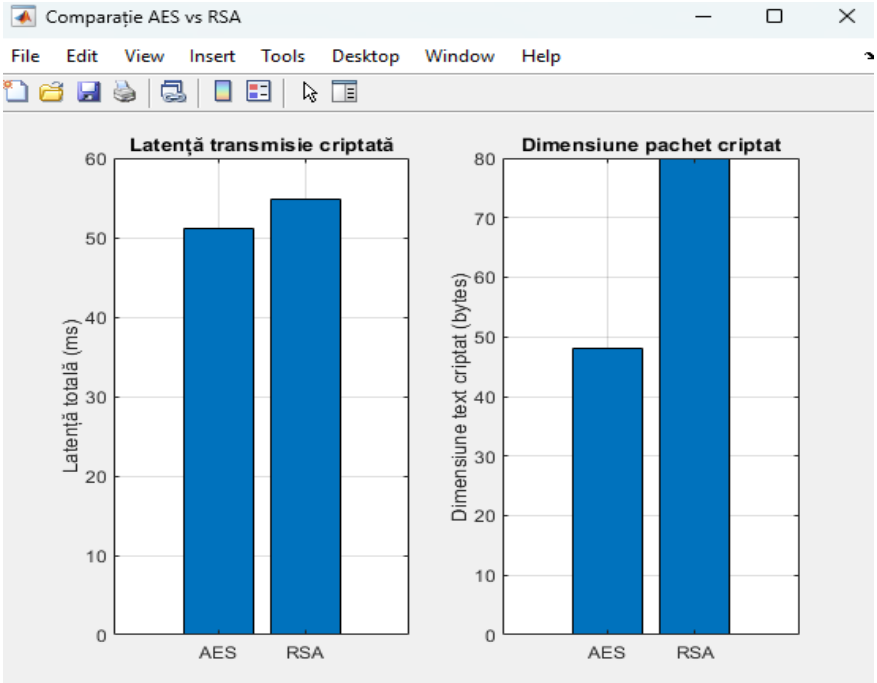

Fig. 9. Comparisons of encrypted transmission latency and encrypted packet size

Total latency (in milliseconds) required for transmitting an encrypted message, including encryption, decryption times, and network delay. The following figure shows that the AES algorithm has lower latency (approximately 52 ms), while the RSA algorithm has higher latency (approximately 56 ms). AES is faster and more efficient in encrypting and decrypting data, resulting in lower overall network latency. RSA, being more complex and involving heavier mathematical operations, introduces greater delays.

Message size (in bytes) after encryption. The AES-encrypted packet is smaller (approximately 48 bytes), whereas the RSA-encrypted packet is significantly larger (approximately 80 bytes). RSA produces a bulkier ciphertext, which can lead to higher bandwidth consumption and an increased likelihood of packet loss in the network. AES, with its more compact size, is better suited for efficient transmission.

Therefore, AES offers an advantageous combination of low latency and compact encrypted data size, making it ideal for continuous data encryption in networks. RSA, while providing a high level of asymmetric security, involves higher latency and larger encrypted data size, making it more suitable for encrypting keys or small messages. The choice of algorithm should depend on the application's requirements: efficiency and speed (AES) versus key exchange security (RSA).

Fig. 10 shows a comparison of the average latency (in milliseconds) for encrypted packets successfully transmitted through the network using the AES and RSA algorithms. The average latency is calculated only for packets that successfully reached their destination (excluding lost packets). It can be observed that both AES and RSA have very similar average latencies, around 50 ms. The difference between them is negligible in this context, suggesting that for successfully transmitted packets, the total processing and transmission time is similar.

The packet delivery success rate is shown in Fig. 11. The success rate is the ratio of correctly received packets to the total number of transmitted packets. Both AES and RSA show a high success rate, around 92-93%. The difference between the two is small, with AES having a slight advantage. The success rate reflects each algorithm's resilience to network losses. Packet size matters: RSA produces larger packets, which can increase the likelihood of loss. Nevertheless, in this simulation, both methods perform almost equally in terms of correctly received packets.

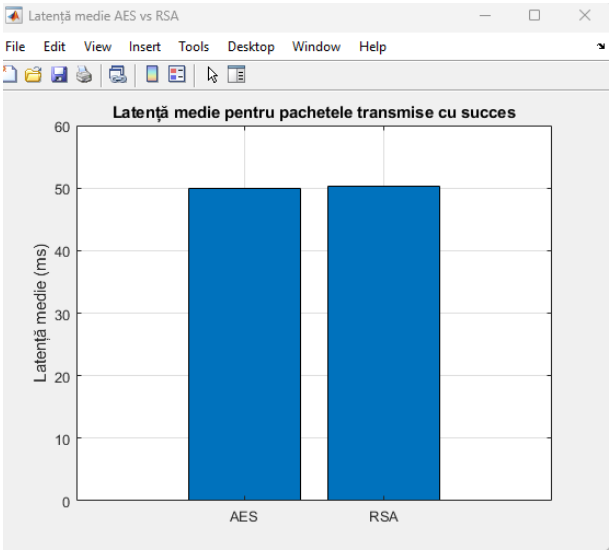

Fig. 10. Average latency for successfully transmitted packets

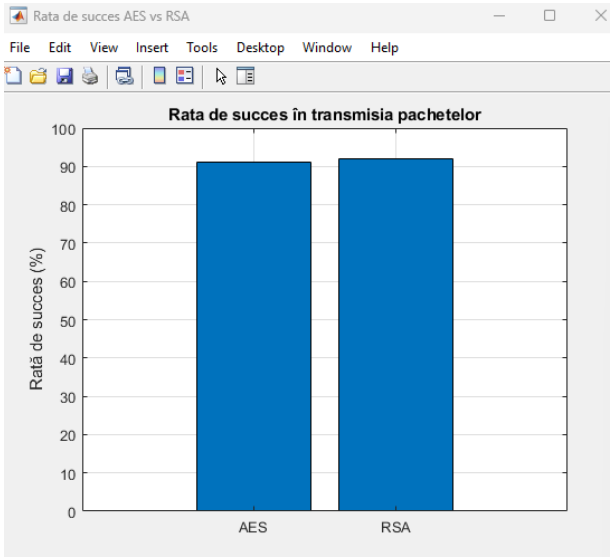

Fig. 11. Packet transmission success rate

The AES algorithm is much more efficient for encrypting large volumes of data, with minimal impact on latency and bandwidth usage. RSA, although slower and producing larger encrypted data, is essential for secure key exchange and authentication. In real networks with latency and packet loss,

AES provides faster and more robust transmission. RSA is used in combination with AES: RSA encrypts the AES key, while AES encrypts the actual data.

5. Conclusions

In this paper, we explored the main encryption methods used to secure communications in computer networks, focusing on two fundamental types of algorithms: AES (Advanced Encryption Standard) - a symmetric encryption algorithm that is highly efficient and fast, and RSA (Rivest-Shamir-Adleman) - an asymmetric encryption algorithm, essential for key exchange and authentication.

We implemented both methods in MATLAB and created simulations that included real network factors such as variable latency and packet loss, in order to evaluate the performance and behavior of these algorithms under conditions as close to reality as possible.

The MATLAB simulations confirmed the theoretical expectations regarding the trade-offs between performance and security in cryptography. Implementing and testing the algorithms in a controlled environment allows for a clear understanding of their operation and their impact on real computer networks. These results provide a solid basis for selecting the optimal encryption method according to the specific requirements of secure communication applications.

References

[1]. Stallings, W. Cryptography and Network Security: Principles and Practice, 7th Edition, Pearson, 2016. ISBN: 9780134444284.

[2]. Kurose, J. F., & Ross, K. W. Computer Networking: A Top-Down Approach, 7th Edition, Pearson, 2017. ISBN: 9780133594140.

[3]. T. Peng, C. Leckie, and K. Ramamohanarao, “Survey of Network-Based Defense Mechanisms Countering the DoS and DDoS Problems,” ACM Computing Surveys, vol. 39, no. 1, pp. 1-42, Apr. 2007.

[4]. MathWorks Documentation - MATLAB Cryptography & Performance Tools, https://www.mathworks.com/help/.

[5]. Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. Handbook of Applied Cryptography, CRC Press, 1996. https://cacr.uwaterloo.ca/hac/.

[6]. Rivest, R. L., Shamir, A., & Adleman, L. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", Communications of the ACM, Vol. 21, No. 2, pp. 120-126, 1978.

[7]. Daemen, J., & Rijmen, V. The Design of Rijndael: AES - The Advanced Encryption Standard, Springer, 2002. ISBN: 9783540425809.