CRYPTOGRAPHIC STRATEGIES FOR PROTECTION AND RECOVERY AGAINST CYBER THREATS

Valentin PAU[1], Constantin-Alin COPACI [2], Dorina-Luminița COPACI[3]

Rezumat. *În era unui proces accelerat de digitalizare, comunicațiile electronice au devenit coloana vertebrală a societății moderne, susținând interacțiunea economică și socială la scară globală. În acest context, criptografia apare nu doar ca instrument tehnologic, ci și ca element fundamental al arhitecturii de securitate. Acest articol își propune să exploreze strategiile criptografice relevante care pot fi aplicate în domeniul comunicațiilor moderne, analizând atât rolul lor în prevenirea breșelor de securitate, cât și contribuția lor la procesele de recuperare și restaurare a încrederii în urma incidentelor cibernetice.*

Abstract. *In the era of an accelerated process of digitalization, electronic communications have become the backbone of modern society, supporting economic and social interaction on a global scale. In this context, cryptography emerges not only as a technological tool but as a fundamental element of security architecture. This article aims at exploring relevant cryptographic strategies that can be applied in the field of modern communications, analyzing both their role in preventing security breaches and their contribution to recovery and trust restoration processes following cyber incidents.*

Keywords: cryptography, cyber threats, data protection, cyber resilience.

DOI 10.56082/annalsarscieng.2025.2.5

1.Introduction

In an increasingly digitalized world, information security has become an essential priority for organizations, governments and individual users. The growing volume of data, system interconnectivity and the complexity of cyberattacks have increased the need for effective protection mechanisms. In this context, cryptography plays a central role in ensuring the confidentiality, integrity and authenticity of data.

In this context, the concept of cyber resilience gains central relevance. Cyber resilience is not limited to preventing attacks but implies a holistic capacity to anticipate, detect, withstand, respond to and recover from security incidents. This involves robust and redundant IT architectures, automated backup and restoration

[1]Professor, PhD, Academy of Romanian Scientists; str. Ilfov nr.3 Bucharest, ZipCode 050044, e-mail: valentin.pau@prof.utm.ro
[2] Engineer, ANCOM, Bucharest;
[3] PhD, Titu Maiorescu University, Bucharest.

processes, strict access and data control policies as well as a set of well-implemented cryptographic mechanisms.

The paper aims at analysing the cryptographic techniques used to protect data as well as at highlighting their role in recovery processes. The paper ends with a set of conclusions obtained from the analyses made in light of the proposed objectives.

2.Cryptography in Cybersecurity

In the context of cybersecurity, cryptography is an essential branch, having as main goal to ensure the confidentiality, integrity, authenticity and non-repudiation of data in digital environments.

By means of advanced mathematical methods and algorithms, cryptography enables the transformation of readable data ("plaintext") into an unintelligible form ("ciphertext"), which can only be understood by authorized entities that hold the appropriate cryptographic keys.

Cryptography is used to protect data both at rest (stored in databases, file systems, or cloud infrastructures) and in transit (transmitted over public or private networks). Additionally, cryptographic mechanisms are indispensable in authentication processes, digital signing and access control.

Modern cryptography relies on four fundamental categories of algorithms and techniques, each having a distinct role in the overall security architecture of information systems.

•*Symmetric encryption* involves using a single key for both encrypting and decrypting data. It is efficient and fast but requires a secure method for key distribution. Common examples include AES (Advanced Encryption Standard) and DES algorithms.

•*Asymmetric encryption* uses a pair of keys: a public key and a private key. The public key encrypts, and the private key decrypts. This model is essential in secure communications and key exchange. Notable examples are RSA and ECC (Elliptic Curve Cryptography).

- *Hash functions* (e.g., SHA-256) generate a fixed summary of a data set. They are used for integrity verification and password storage.

•*Digital signatures* combine asymmetric encryption and hash functions to ensure the authenticity and integrity of a message or document.

The security of any cryptographic system depends on the effective protection of cryptographic keys. Poor management can compromise the entire system, irrespective of the strength of the used algorithm.

It is essential to implement solutions such as HSM (Hardware Security Modules), to change keys periodically, to have clear access policies and to store keys in isolated environments.

3.Case Study - Cyberattacks on Digital Signatures and Preventive Measures Using Cryptographic Techniques

Electronic signature, as an extension of modern cryptographic technologies, is essential for ensuring the authenticity and integrity of information in electronic communications. However, its efficiency can be undermined by a series of cyberattacks targeting either the cryptographic mechanism itself or its implementation.

Attacks on electronic signatures, such as private key compromise or signature forgery, can have serious consequences, including financial losses and damage to reputation. Therefore, integrating robust cryptographic strategies as well as monitoring and recovery mechanisms are crucial to ensure the resilience of information systems against these threats.

3.1.Brute Force Attack

In the case of this type of attack, an attacker tries all possible combinations of private keys to find the key which was used for the digital signature. Brute force attacks are ineffective if large-sized keys are used (for example, 2048 bits for RSA or 256 bits for ECDSA), as the number of possibilities is extremely large. However, if shorter or weaker keys are used, the attack may be feasible.

Here are some protective measures that can be taken to prevent such attacks:

- The use of a strong private key

-*Large key lengths*: An appropriate key length should be used for digital signatures. For example, 2048 bits is considered the minimum for RSA, while 4096 bits provides an additional level of security. Shorter keys are more vulnerable to brute force attacks.

An OpenSSL command to generate a 4096-bit RSA key: *openssl genpkey -algorithm RSA -out private_key.pem -pkeyopt rsa_keygen_bits:4096*. After running this command, OpenSSL will generate a 4096-bit RSA private key and

save it in the file private_key.pem. This private key can be used for digital signing or for decrypting data encrypted with the corresponding public key (Figure 1).

At the same time, the generated private key can also be used to create an associated public key, which can be distributed to other users to encrypt data intended to be decrypted only by the holder of the private key.

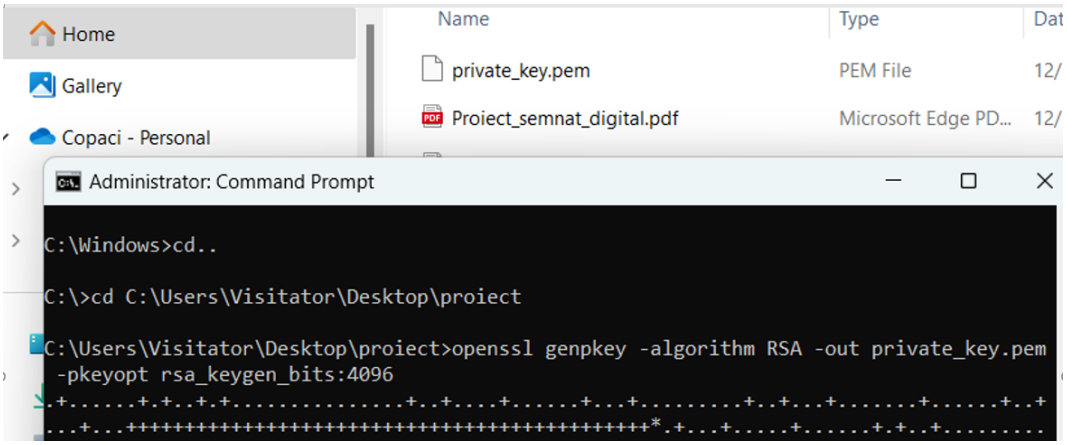

Fig. 1. Generating a 4096-bit RSA key

- *ECC keys (Elliptic Curve Cryptography)*: If a higher level of security with a smaller key length is desired, ECC (Elliptic Curve Cryptography) provides a more efficient encryption, which is more resistant to attacks than RSA. A 256-bit ECC key is considered equivalent to a 3072-bit RSA key.

The OpenSSL command for generating a 256-bit ECC private key using the prime256v1 elliptic curve is: *openssl ecparam -name prime256v1 -genkey -noout -out private_key.pem* will create a file containing the private key, which can be used for elliptic curve cryptography (Figure 2). The file private_key.pem will contain the private key in PEM (Privacy-Enhanced Mail) format, a text-based format using Base64 encoding commonly employed to store cryptographic keys and certificates. The private key will include information about the elliptic curve used and its associated parameters, but this piece of information will not be displayed in the terminal due to the use of the *-noout* option.

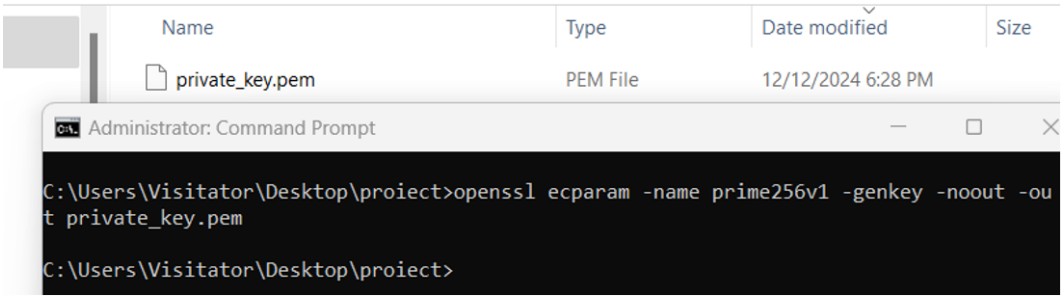

Fig. 2. Generating the ECC private key (256 bits) using the prime256v1 elliptic curve

After generating the private key, the associated public key can be extracted using the command: *openssl ec -in private_key.pem -pubout -out public_key.pem* (Figure 3).

This will save the public key in a file named *public_key.pem*.

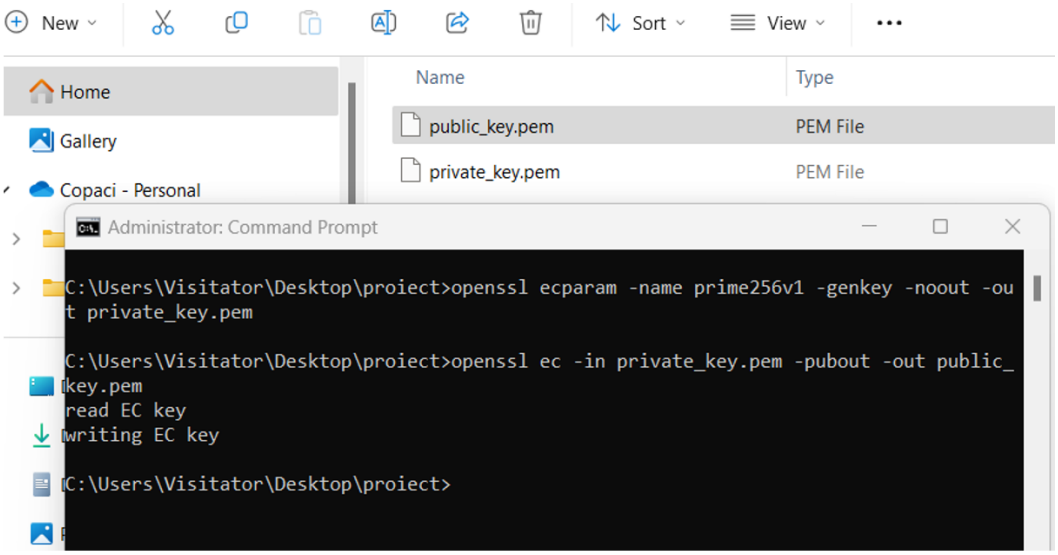

Fig. 3. Public key extraction

➢Encrypting and Protecting Private Key

The private key used for the digital signature must be protected with a strong password to prevent unauthorized access. This can prevent an attacker from accessing the private key even if he/she manages to obtain it.

The encryption of the private key with OpenSSL is done using the command: *openssl genpkey -algorithm RSA -out private_key.pem -aes256 -pkeyopt rsa_keygen_bits:4096* (Figure 4)

Thus, a 4096-bit RSA private key will be generated. It will be encrypted with AES-256 and saved in a PEM file (private_key.pem). Protecting the private key with AES-256 adds an additional layer of security, ensuring that the key cannot be used without the correct password.

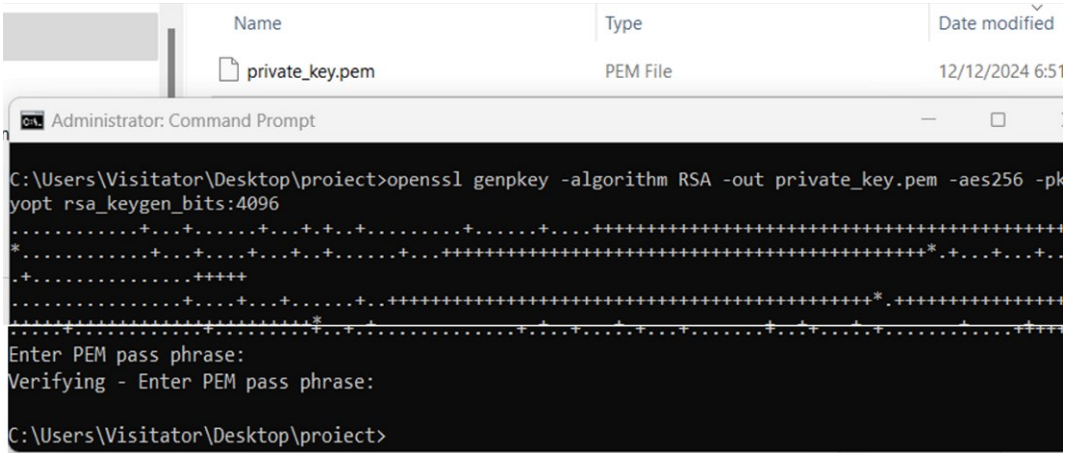

Fig. 4. Encrypting private key with OpenSSL

If the encrypted private key is to be used for cryptographic operations, OpenSSL will prompt for the encryption password.

For example, if one wants to obtain the decrypted key from the file private_key.pem, one can use the command: *openssl rsa -in private_key.pem -out decrypted_key.pem*, which will save the decrypted private key in the file *decrypted_key.pem* (Figure 5).

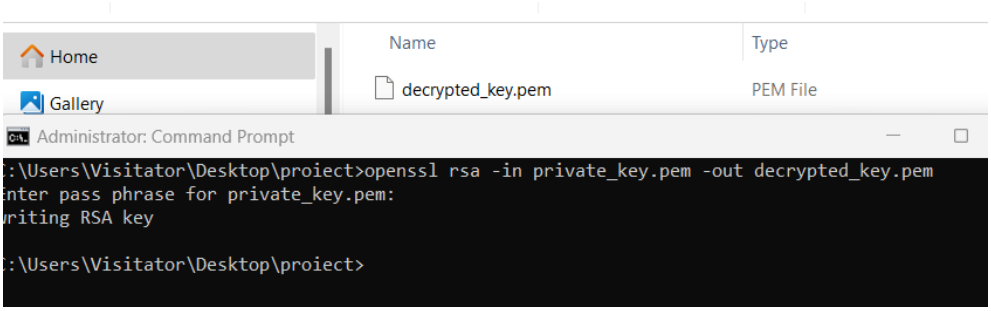

Fig. 5. Obtaining the decrypted key from the file *private_key.pem*

➢Protection against brute force attacks on the SSL/TLS protocol

If a digital signature is used in a security protocol such as TLS (for example, for HTTPS), it is important to ensure that the server and client use recent versions of TLS (TLS 1.2 or TLS 1.3). Older versions (SSL 2.0/3.0) are vulnerable to various attacks.

➢Monitoring and detecting unauthorized activities

- Monitoring access to private keys: Implementing a logging and monitoring system for accessing private keys or digital signatures. This can help quickly identify unauthorized attempts to access or use the private key.

- Using intrusion detection systems (IDS): An IDS can help identify some attacks that try to obtain the private key or use it unauthorizedly.

3.2.Replay Attack

In a replay attack, an attacker captures a valid signed message and retransmits it to another party, hoping that the recipient will accept the original signature as valid. If the digital signature does not include information that binds the signature to a specific session, time or other unique characteristics, an attacker can retransmit the signature to achieve a desired outcome. It is essential that the digital signature be associated with a timestamp or a unique transaction identifier (for example, a serial number or a nonce) so that the signature cannot be reused.

OpenSSL can be used to generate and verify digital signatures, and the protection of signatures can be strengthened by using a secure channel and mechanisms for revoking signatures or certificates.

The command *openssl dgst -sha256 -sign private_key.pem -out proiect_atac.sig proiect_atac.txt* will sign a message with the private key. The command *openssl dgst -sha256 -verify public_key.pem -signature proiect_atac.sig proiect_atac.txt* verifies the signature using the public key generated from the same private key (Figure 6):

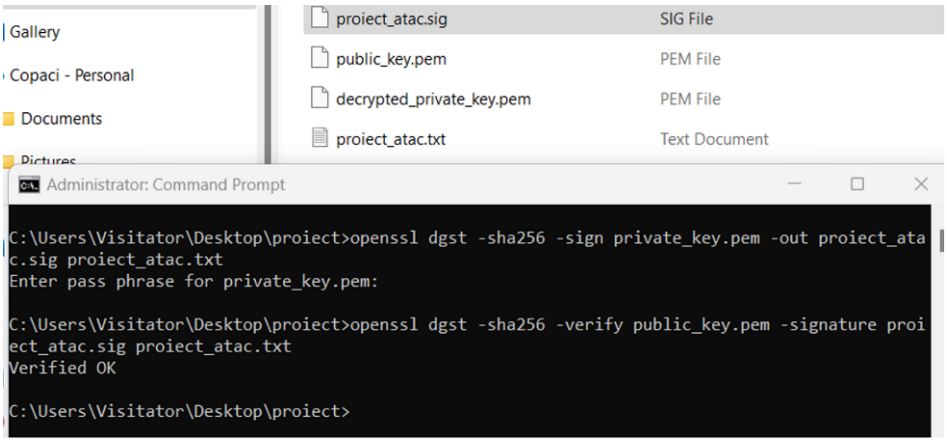

Fig. 6. Checking the signature using the public key generated from the same private key

3.3.Hash Collision Attack

In a hash collision attack, an attacker looks for two different messages that generate the same cryptographic digest (hash). If this is possible, the attacker can create a different document (but with the same hash) that is validated with the signer's digital signature.

A collision attack can compromise the integrity of the digital signature because a valid signature on one document could be used for another different document.

The use of strong and collision-resistant hash algorithms, such as SHA-256 or SHA-3, are considered secure against this type of attack.

One of the best-known cases of hash collision attacks was the attack on the SHA-1 hash function. In 2017, researchers from Google and the University of Copenhagen demonstrated a collision attack on SHA-1, leading to its gradual deprecation. The attack was carried out on the SHA-1 function, which is a cryptographic hash algorithm widely used for digital signatures, SSL/TLS certificates and many other applications. In 2017, the team of researchers demonstrated for the first time that they could generate a SHA-1 hash collision in less time than previously considered possible. This was achieved through an attack called SHAttered.

Researchers used advanced computing techniques and optimization methods to create two different documents that had exactly the same SHA-1 hash value. They achieved this collision with a computational cost which was significantly lower

than previously thought, thus reducing the time needed to find a collision. This attack had a direct impact on the trust in SHA-1 as a secure algorithm for cryptographic applications.

Hash collisions allow an attacker to:

-modify digitally signed documents or files while keeping the original signature intact.

- create fraudulent SSL/TLS certificates with the same hash values as legitimate certificates, compromising the security of websites using those certificates.

Basically, the attack demonstrated that, for security purposes, SHA-1 can no longer be considered a secure algorithm to be used in digital signatures and in other critical security applications.

After the 2017 attack, many organizations and security authorities have recommended the urgent replacement of the SHA-1 function with more secure hash functions, such as SHA-256. For example, Google Chrome and Mozilla Firefox began to no longer accept SSL/TLS certificates that use SHA-1.

Certification authorities (CAs) have begun phasing out the issuance of SSL/TLS certificates based on SHA-1. Many of these certificates have been revoked or replaced with others based on more secure algorithms, such as SHA-256.

Therefore, including some concrete cases about attacks on electronic signatures can enrich the paper, making it more relevant and applied.

4.Conclusions

In an increasingly complex digital landscape, cryptography is no longer just an added layer of security but a fundamental necessity. When applied correctly, it protects critical data, ensures the integrity of communications and facilitates recovery processes after cyberattacks

Cyber resilience cannot exist without a solid, well-managed cryptographic framework which is adapted to new challenges. It is imperative for organizations to implement strict key management policies, use modern cryptographic technologies and plan for the transition to post-quantum cryptography.

Electronic signature is an indispensable tool for authenticating and validating digital transactions. Attacks on digital signatures can compromise the security of a cryptographic system, but there are preventive measures and protective techniques

to mitigate risks. The most important thing is to use strong cryptographic algorithms, protect private keys and implement security protocols to protect the integrity and authenticity of digital signatures.

The future of cybersecurity will largely depend on how cryptography will evolve and will be integrated into the defence and recovery strategies of organizations.

R E F E R E N C E S

[1] Katz, Jonathan; Lindell, Yehuda. Introduction to Modern Cryptography, Revised Third Edition. Chapman & Hall / CRC Press, 2025;

[2] Easttom, William. Modern Cryptography: Applied Mathematics for Encryption and Information Security (ediția a II-a), Springer, 2022;

[3] Banoth, Rajkumar; Regar, Rekha. Classical and Modern Cryptography for Beginners. Springer Cham, 2023;

[4] https://www.tutorialspoint.com/cryptography/sha_1_algorithm.htm

[5] Stallings, W. (2017). *Cryptography and Network Security: Principles

[6] Goldreich, Oded (2004), Foundations of cryptography II: Basic Applications (ed. 1. publ.), Cambridge [u.a.]: Cambridge Univ. Press, ISBN 978-0-521-83084-3;

[7] Howgrave-Graham, N. A., & Smart, N. P. (2001). Lattice Attacks on Digital Signature Schemes. Designs, Codes and Cryptography, 23, 283–290.

[8] Li, J. (2023). A research on digital signature schemes. Applied and Computational Engineering, 16, 65-74

[9] Irimia, A.-V. (2023). Ransomware data recovery techniques. Cryptology ePrint Archive, Paper 2023/125

[10] Zimba, A., Wang, Z., Simukonda, L. (2018). Towards Data Resilience: The Analytical Case of Crypto Ransomware Data Recovery Techniques. International Journal of Information Technology and Computer Science (IJITCS), Vol.10, No.1

[11] Quantum Computing, Cyber Security and Cryptography: Issues, Technologies, Algorithms, Programming and Strategies. Goyal, S.B., Kumar, Vidyapati, Islam, Sardar M.N., Ghai, Deepika (Eds.). (2025). Springer.