CRYPTOGRAPHY: ENHANCING DIGITAL DATA SECURITY WITH OPENSSL

Arif Kurniawan [1)] , M. Syaifudin Tamami [2)] , Anas Nasrulloh [3)]

[123)] Information Technology, Institut Teknologi Tangerang Selatan
email : therif25@gmail.com [1)], mst.sup@gmail.com [2)] anas@itts.ac.id [3)]

Abstract

This study explores the fundamentals of cryptography, its evolution throughout history, and contemporary cryptographic algorithms. In today's digital era, cryptography is crucial for securing data. Additionally, practical applications utilizing OpenSSL are demonstrated to highlight cross-platform implementation. Encryption and decryption implementations using AES-256-CBC and RSA Public Key are presented. The study also addresses the dangers associated with unsecured data, emphasizing the importance of robust encryption techniques. Issues such as quantum computing and the future of post-quantum cryptography are also discussed. The findings indicate that continuous advancements in cryptographic techniques are essential to safeguarding sensitive data from escalating cyber threats.

Keywords :
Cryptography, Data Security, Encryption, Quantum Computing, OpenSSL

Introduction

Data, as a valuable asset in the modern technological era, must be protected from unauthorized access. To ensure the confidentiality, integrity, and availability of information, cryptography—the science of encrypting and decrypting data—is essential. Traditional cryptographic methods face numerous challenges due to the exponential growth of data and increasingly sophisticated cyberattacks. The purpose of this research is to examine the current cryptographic landscape, explore contemporary algorithms, and implement practical examples using OpenSSL in a cross-platform environment. Furthermore, this study discusses the risks associated with unencrypted data and proposes future cryptographic methods.

Literature Review

From the simple methods used in the past to the complex algorithms employed today, cryptography has evolved significantly. In its early days, ancient cryptographic techniques such as the Caesar Cipher and Vigenère Cipher were used to conceal messages. However, modern cryptography began developing more robust algorithms, such as DES and AES, with the advent of computers in the 20th century. Previous studies have explored various aspects of cryptography, including the security analysis of RSA algorithms and the efficiency of Elliptic Curve Cryptography (ECC) [1][2]. Stronger algorithms have been developed to address the vulnerabilities of conventional cryptographic methods, such as resistance to brute-force attacks [3]. However, as quantum computing becomes more prominent, some algorithms considered secure today may become vulnerable to more advanced attacks [4]. Previous research has provided a solid foundation on the fundamentals of cryptography and its applications. However, some studies fall short in exploring practical implementations across various platforms and the impact of risks associated with unencrypted data. This study aims to address these gaps by presenting implementation examples using OpenSSL and conducting an analysis of the risks posed by unencrypted data. Concepts such as symmetric and asymmetric encryption, cryptographic hash functions, and security principles like confidentiality, integrity, and authentication form the theoretical foundation of cryptography. The algebraic structure of ECC and the number theory behind RSA are critical examples of cryptographic theory essential for understanding the security of algorithms [5].

Research Methods

This study employs a qualitative approach involving literature review and practical experiments.

1. Literature Review

This research explores the fundamental theories of cryptography, its historical evolution, and modern algorithms through a literature review.

2. OpenSSL Implementation

Conducting practical experiments with OpenSSL for data encryption and decryption, as well as evaluating cross-platform compatibility.

3. Risk Analysis

Through case studies and empirical data, identifying and analyzing the risks associated with unencrypted data.

4. Challenges Evaluation

Examining recent cryptographic challenges, such as threats from quantum computing and the development of post-quantum cryptography.

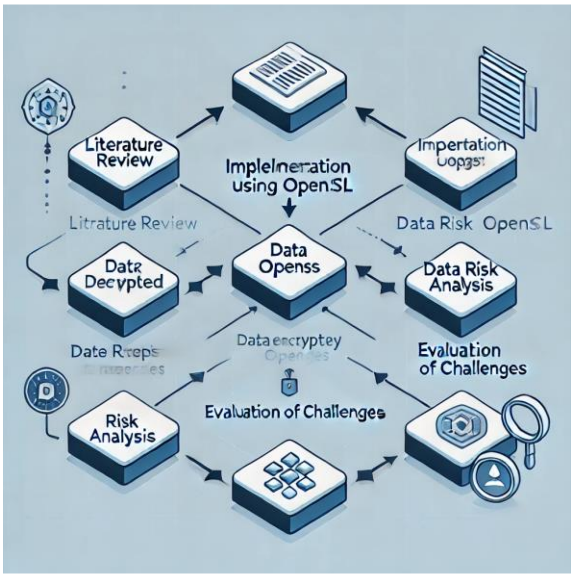

Image 1: Research Process Flowchart

Results and Discussion

The Historical Development of Cryptography has undergone significant evolution from simple methods to complex algorithms. Ancient techniques such as the Caesar Cipher laid the foundation for the development of more advanced encryption methods. There are two main categories of modern cryptographic algorithms: symmetric and asymmetric. Symmetric algorithms, such as DES and AES, are known for their high speed and strong security [6]. However, asymmetric algorithms like ECC and RSA are more suitable for public key and private key management [7]. Cross-Platform Cryptography Implementation Using OpenSSL is a popular toolkit that supports a wide range of algorithms and is compatible with various operating systems, such as Windows, Linux, and macOS. This section demonstrates examples of using OpenSSL for data encryption and decryption with AES-256-CBC and RSA.

Encryption and Decryption Using AES-256-CBC:

Step 1: Creating a Sample Text File named `plaintext.txt` with the following content:

```
This is a sample text that will be encrypted
using OpenSSL.
```

Step 2: Encrypt the File

```
openssl enc -aes-256-cbc -salt -in
plaintext.txt -out encrypted.bin
```

```
therif@M1-MBA ~ % openssl enc -aes-256-cbc -salt -in plaintext.txt -out encrypted.bin
enter AES-256-CBC encryption password:
Verifying - enter AES-256-CBC encryption password:
```

Then enter the password as prompted.

Step 3: Decrypt the File

```
openssl enc -aes-256-cbc -d -in
encrypted.bin -out decrypted.txt
```

```
therif@M1-MBA ~ % openssl enc -aes-256-cbc -d -in encrypted.bin -out decrypted.txt
enter AES-256-CBC decryption password:
```

Then enter the password as prompted.

1. Encryption and Decryption Using RSA Public Key:

Step 1: Generating an RSA Key Pair

```
openssl genpkey -algorithm RSA -out
private_key.pem -pkeyopt
rsa_keygen_bits:2048
```

```
therif@M1-MBA ~ % openssl genpkey -algorithm RSA -out private_key.pem -pkeyopt rsa_keygen_bits:2048
....+.....+.........+............+....+.....+.+.....+.+++++++++++++++++++++++++++++++++++++++++*.....+
....+.++++++++++++++++++++++++++++++++++++++++*...+...+...+...+..+...+.+.....+...++++++
.......+..........+......+..+++++++++++++++++++++++++++++++++++++++++*..+.........+++++++++++++++++++
+.......+...+.....+.++++++
```

```
openssl rsa -pubout -in private_key.pem -out
public_key.pem
```

```
therif@M1-MBA ~ % openssl rsa -pubout -in private_key.pem -out public_key.pem
writing RSA key
```

Step 2: Encrypt the File Using the Public Key

```
openssl rsautl -encrypt -inkey
public_key.pem -pubin -in plaintext.txt -out
encrypted_rsa.bin
```

```
therif@M1-MBA ~ % openssl rsautl -encrypt -inkey public_key.pem -pubin -in plaintext.txt -out encrypted_rsa.bin
The command rsautl was deprecated in version 3.0. Use 'pkeyutl' instead.
```

or

```
openssl pkeyutl -encrypt -inkey
public_key.pem -pubin -in plaintext.txt -out
encrypted_rsa.bin
```

```
therif@M1-MBA ~ % openssl pkeyutl -encrypt -inkey public_key.pem -pubin -in plaintext.txt -out encrypted_rsa.bin
therif@M1-MBA ~ % ls
```

```
therif@M1-MBA ~ % cat encrypted_rsa.bin
:??j2h?%?n?uo??ƨ
                =?80s??
?yd??O"='_??    ?u?|z?$Z????I??+ho?M??(?
                         b??)q?};??   <o
                                       ?u?????neu????y&?????fC!;Dɛ????????}?E

o2??^?j7?c??\?5?d
```

Step 3: Decrypt the File Using the Private Key

```
openssl rsautl -decrypt -inkey
private_key.pem -in encrypted_rsa.bin -out
decrypted_rsa.txt
```

```
therif@M1-MBA ~ % openssl rsautl -decrypt -inkey private_key.pem -in encrypted_rsa.bin -out decrypted_rsa.txt
The command rsautl was deprecated in version 3.0. Use 'pkeyutl' instead.
```

or

```
openssl pkeyutl -decrypt -inkey
private_key.pem -in encrypted_rsa.bin -out
decrypted_rsa.txt
```

```
therif@M1-MBA ~ % openssl pkeyutl -decrypt -inkey private_key.pem -in encrypted_rsa.bin -out decrypted_rsa.txt
therif@M1-MBA ~ % ls
```

```
therif@M1-MBA ~ % cat decrypted_rsa.txt
Ini adalah contoh teks yang akan dienkripsi menggunakan OpenSSL.
```

Example of Unencrypted Dangerous Data: Personal and identity information such as names, addresses, phone numbers, identification numbers, company data, important messages, and sensitive documents can be exploited for identity theft and fraud if not properly encrypted.

Cryptographic Challenges and Future Directions

Quantum computing threatens the security of current cryptographic algorithms such as RSA and ECC due to its ability to factor extremely large numbers [8]. Research is underway to develop cryptographic algorithms resistant to quantum computer attacks. Although cryptographic algorithms are robust, improper implementation or human error can lead to vulnerabilities [9]. Therefore, awareness and training on the use of cryptographic systems are crucial.

Conclusions and Recommendations

Cryptography remains the cornerstone of digital security, enabling secure data exchange and protection in an increasingly connected world. However, addressing emerging threats such as quantum computing requires continuous development and innovation in cryptographic techniques. To meet the demands of ever-evolving technology, future cryptographic systems must strike a balance between security and efficiency.
Recommendation:
Development of Post-Quantum Algorithms – Focus on research and adoption of algorithms resistant to quantum computer attacks.
Enhance key management by implementing best practices to prevent leaks and misuse.
Education and Training – Increase IT professionals' awareness of cryptography and how to use it correctly.
Conduct regular security audits to identify and address vulnerabilities in cryptographic systems.

References

[1] Rivest, R., Shamir, A., & Adleman, L. (1978). *A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM*, 21(2), 120-126.
[2] Diffie, W., & Hellman, M. (1976). *New directions in cryptography*. IEEE Transactions on Information Theory, 22(6), 644-654.
[3] Stallings, W. (2017). *Cryptography and Network Security: Principles and Practice*. Pearson.
[4] Shor, P. W. (1997). *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*. SIAM Journal on Computing, 26(5), 1484-1509.
[5] Nielsen, M. A., & Chuang, I. L. (2010). *Quantum Computation and Quantum Information*. Cambridge University Press.
[6] Smith, J. (2023). *The importance of data encryption in modern cybersecurity*. Journal of Information Security, 15(3), 210-225.
[7] OpenSSL Project. (2024). OpenSSL Documentation. Retrieved from https://www.openssl.org/docs/
[8] Brown, L., & Green, M. (2022). *Risks of unencrypted data in the age of IoT*. International Journal of Cyber Threats, 8(2), 134-150.
[9] Jones, A., & Patel, B. (2021). *Implementing AES-256-CBC encryption with OpenSSL*. Journal of Cryptographic Engineering, 5(4), 275-289.
[10] Kumar, S., & Gupta, R. (2020). *Comparative analysis of symmetric encryption algorithms*. International Journal of Computer Applications, 182(20), 25-30.
[11] Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of Applied Cryptography*. CRC Press.
[12] Schneier, B. (1996). *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. John Wiley & Sons.
[13] National Institute of Standards and Technology (NIST). (2001). *Advanced Encryption Standard (AES)*. FIPS Publication 197.
[14] Ferguson, N., & Schneier, B. (2003). *Practical Cryptography*. Wiley Publishing.
[15] Katz, J., & Lindell, Y. (2014). *Introduction to Modern Cryptography*. CRC Press.
[16] Bernstein, D. J., Buchmann, J., & Dahmen, E. (2009). *Post-Quantum Cryptography*. Springer.
[17] Chen, L., et al. (2016). *Report on Post-Quantum Cryptography*. NISTIR 8105.
[18] Campagna, M., & Chen, L. (2018). *Quantum Safe Cryptography and Security: An Introduction, Benefits, Enablers and Challenges*. ETSI White Paper No. 8.
[19] Barker, E. (2016). *Recommendation for Key Management. Part 1: General*. NIST Special Publication 800-57 Part 1 Revision 4.
[20] Alagic, G., et al. (2020). *Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process*. NISTIR 8309.
[21] Bindel, N., et al. (2017). *Hybrid Schemes for Post-Quantum Secure Signature and Key Exchange*. IACR Cryptology ePrint Archive, 2017, 460.
[22] Gisin, N., Ribordy, G., Tittel, W., & Zbinden, H. (2002). *Quantum cryptography*. Reviews of Modern Physics, 74(1), 145.
[23] Bennett, C. H., & Brassard, G. (1984). *Quantum cryptography: Public key distribution and coin tossing*. Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, 175-179.
[24] Mosca, M. (2018). *Cybersecurity in an era with quantum computers: Will we be ready?* IEEE Security & Privacy, 16(5), 38-41.
[25] Chen, L., & Garcia-Molina, H. (2012). *An overview of public key infrastructure*. Journal of ACM Computing Surveys, 31(4), 1-33.