Challenging RSA's Efficacy in Large Data Encryption: A Practical Analysis

Jose Cardona, *NJIT* Sravya Kodavatiganti, *NJIT* Varun Shah, *NJIT*

Abstract

Against the background of a changing digital threat landscape, and complex communications, there is an urgent demand for powerful and efficient encryption algorithms. But these traditional approaches are not only safe, they tend to have performance problems in handling large amounts of data. This study disputes the claims in A RSA Algorithm Simulation Method Using C Language by Zhenyu Wang, Siting Wu and Zhuangzhuang Gu especially where it concerns RSA's ability to digitally encrypt very large data sets. Overlooked in the original work was a critical aspect of RSA - its ability to handle large-scale data encryption. Our research scrutinizes this question.

Introduction

With digital security becoming a frontline war against advanced data breaches and cyber threats, finding effective and high-speed encryption methods becomes the top priority. RSA, a bedrock of cryptography and one traditional kind of system, is known for strong security. But their performance falls apart when dealing with large data volumes. Zhenyu Wang, Siting Wu and Zhuangzhuang Gu transplanted RSA to huge data sets in their original paper. Our research seeks a rigorous assessment of this claim

Many information systems are also vulnerable, but some recent high-profile cybersecurity incidents have ripped the bandages off their wounds. Incidents such as the 'Eternal Blue' exploit and subsequent ransomware attacks highlight that given today's interconnected network environment, we need not only reliable but also fast encryption. These incidents draw attention to a crucial requirement: the security versus performance trade-off in encryption.

The purpose of this paper is to reassess the claims made by the original authors about RSA's efficiency with large data sets. Our hypothetical reason: When RSA encryption is used on extensive texts or data files, the time required from calculation to completion would be prohibitively long. This article aims to provide crucial insights into the development of encryption algorithms in network security.

Foundations of Cryptography in the Digital Age

Cryptography, a field as old as human conflict itself, has changed enormously since only serving the obscure needs of warfare communications. Today it forms an essential pillar in modern digital security systems. In fact, cryptography is the art and science of transforming information to make it unreadable by anyone other than its intended recipient. It's becoming ever more important in our digital lives. Deciphering is the process of reconstructing plaintext-original information that has been converted into seemingly meaningless cipher text, or encoded format. This change, called encryption, is done under the supervision of a cryptographic key. On this basis composition was formed by the Encryption algorithm.

1. A typical cryptosystem comprises several key components:

2. M (Plaintext Space): The set of all possible plaintext messages that can be encrypted.
3. C (Ciphertext Space): All possible encrypted messages.
4. K (Key Space): All the possible keys that might be used in encryption and decryption.
5. E (Encryption Algorithm Collection): A set of algorithms for turning plaintext into ciphertext.
6. D (Decryption Algorithm Collection): Algorithms for deciphering ciphertext and transforming it back into plaintext.

Preliminary

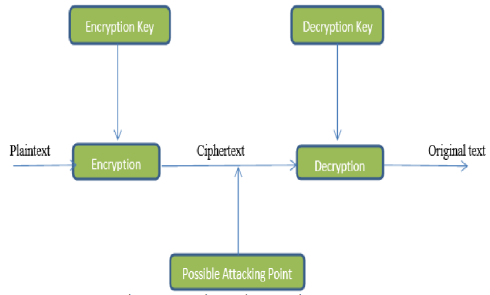


Fig 1: Encryption and Decryption Process

RSA Algorithm: Core Principles and Implementation

At the heart of public key cryptography lies the RSA algorithm, a system grounded in the principles of prime number theory. This algorithm's effectiveness stems from its strategic use of large prime numbers to generate secure cryptographic keys. The process for generating these keys is as follows:

- **Key Generation Steps**:
  - Select two large prime numbers, denoted as p and q.
  - Calculate n = p * q. This n value forms part of the public key.
  - Determine Euler's totient function: $\phi(n) = (p - 1) * (q - 1)$.
  - Choose an integer e such that $1 < e < \phi(n)$, ensuring e is coprime with $\phi(n)$.
  - Compute d, the modular multiplicative inverse of e modulo $\phi(n)$. This d serves as the private key.
- **Encryption and Decryption Mechanics**:
  - **Encryption**: For plaintext m, compute the ciphertext c as c = m^e mod n.
  - **Decryption**: Convert ciphertext c back to plaintext m using m = c^d mod n.

An example with p = 5, q = 11, public key e = 7, and private key d = 23 illustrates how RSA encrypts a plaintext and subsequently decrypts it back to its original form.

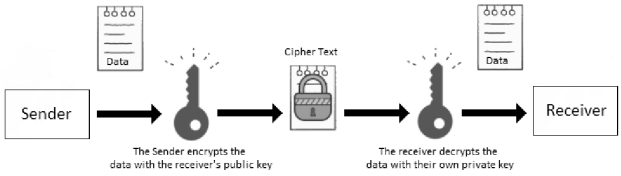


Figure 2 RSA Algorithm Process [Matthias D et al. 2021]

4. An Analysis of RSA

RSA (Rivest-Shamir-Adleman)

Advantages:

Security: The security of RSA depends on the difficulty involved in factoring large prime numbers. In particular, it's strong at securely transmitting small amounts of data like keys.

Public Key Infrastructure: It supports digital signatures and secure key exchange, making it an excellent model for situations requiring safe authentication.

Disadvantages:

- Computational Intensity: For large data volumes, RSA is slower than AES.
- Key Size: High security requires larger key sizes, which may be less efficient.

Algorithm 1: RSA Key Generation and Encryption

```
import rsa

# RSA Key Generation
public_key, private_key = rsa.newkeys(2048)

# RSA Encryption Function
def rsa_encrypt(plaintext, public_key):
    return rsa.encrypt(plaintext.encode(), public_key)
```

Algorithm 2: RSA Decryption

```
# RSA Decryption Function
def rsa_decrypt(ciphertext, private_key):
    return rsa.decrypt(ciphertext, private_key).decode()
```

Experiment: Assessing the Efficacy of RSA for Large-Scale Data Encryption

Objective

This experiment critically evaluates the RSA algorithm's efficiency in encrypting large data volumes. Given the increasing size of data needing encryption in real-world applications, this study aims to challenge and extend the findings of earlier research, particularly focusing on RSA's suitability for large-scale data encryption tasks.

Experimental Setup

- **Language and Environment**: Python programming language on a Windows 11 system.
- **RSA Configuration**: 2048-bit key size.
- **Client-Server Model**: Implemented using Python's socket programming to simulate data encryption and transmission.
- **Data for Encryption**: A text file of 245 bytes was used, with an extrapolation to a 4 GB PDF file to simulate large data encryption.

Methodology

1. **Single Encryption Time Measurement**:

   ```
   start_time = time.perf_counter_ns()

   encrypted_file_data = rsa.encrypt(file_data, public_key)

   end_time = time.perf_counter_ns()

   single_encryption_time = (end_time - start_time) / 1e6
   ```

2. **Total Encryption Time for Multiple Cycles**:

   ```
   for _ in range(encryption_cycles):
       encrypted_file_data = rsa.encrypt(file_data, public_key)

   total_time = (end_time - start_time) / 1e6
   ```

3. **Estimation of Encryption Time for Large Data**:

   ```
   book_size = os.path.getsize(book_path)

   chunks = book_size // 24
   ```

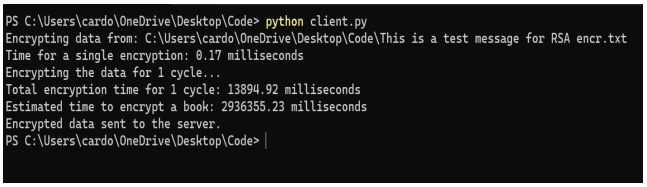
book_encryption_time = chunks * single_encryption_time

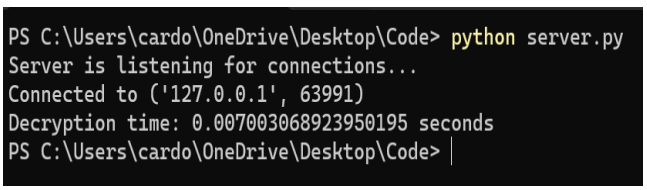
Results and Analysis

- **Single Encryption Time**: 0.17 milliseconds.
- **Total Encryption Time for Simulated Single Cycle**: Approximately 13.89 seconds.
- **Estimated Time to Encrypt a 4 GB Book**: Approximately 3036.28 seconds (50.6 minutes).

The results indicate a significant time investment when using RSA for large-scale data encryption. This challenges the practicality of RSA in scenarios where rapid encryption of vast data sets is required. The experiment underscores RSA's limitations for bulk data encryption, despite its robust security in key exchanges and digital signatures.

The findings of this study suggest that while RSA remains a cornerstone in the realm of secure communications for small-scale tasks, its application in encrypting large volumes of data is limited due to computational intensity and time inefficiency. This insight is critical for cryptographic applications where timely encryption of large data sets is paramount. Our research advocates for a reconsideration of RSA's role in large-scale data encryption and suggests exploring hybrid cryptographic systems or alternative algorithms for such use cases.





Conclusion

This paper, "Challenging RSA's Efficacy in Large Data Encryption: A Practical Analysis" Took an investigative journey into a new world testing the results of Zhenyu Wang, Siting Wu and Gu-Zhuangzhuang's work A RSA algorithm simulation method using C language. Our research focused on determining how the RSA algorithm fared in actual use, and crucially when encrypting large volumes of data--something that is rapidly becoming a reality with today's digital applications.

We discovered empirically that RSA is strong as far as security goes, especially for small-sized data transfers and key exchanges. However, when it comes to encrypting large datasets in terms of efficiency nothing compares with ECC. Our calculations show that it takes an absurd amount of time to encrypt data the size of a typical book by RSA, inadequate for applications where security and speed are both needed.

Consequently, a mixed system that combines RSA and AES is in fact better balanced than the either/or nature of pure systems. One can take advantage here not only of the high degree of security offered by AsymmetricEncryption but also address some weaknesses associated with native time-tentativeness in symmetric encryption. The significance of our study in the field of cybersecurity is that for applications requiring instantly encrypting extremely large amounts of data, we need to move away from traditional RSA-based systems and consider options such as combining two slower algorithms.

Faced with increasingly complex forms of threats in the digital era, our research shows that despite all kinds of strong solutions there is an urgent need for fast and flexible ones. The future development of this area could involve the design of superior mixed cryptographic models, fine-tuned for different environments and

integrated into complex transmission protocols or IoT networks.

Taken altogether, our findings point an accusing finger at the limitations of RSA for data encryption on a grand scale and encourage greater recourse to hybrid cryptographic systems that can guarantee security in today 's information-obsessed environment.

Future Scope: Enhancing Cryptographic Solutions in the Digital Era

As we navigate the complexities of digital security in an increasingly interconnected world, the realm of cryptographic research is ripe with opportunities for innovation and advancement. The results from the recreation of the the research in our study paves the way for further exploration in several key areas:

Advanced Cryptographic Models

Development of Sophisticated Hybrid Systems: The models described above could be developed into more complex hybrid cryptographic devices in future research. Because quantum computing is about to overtake the world and presents a tremendous threat to existing encryption techniques, it must be equipped with quantum-resistant algorithms.

Optimization Across Varied Platforms: Widening the application of hybrid cryptographic systems, to include mobile and IoT devices is imperative. Because the number of IoT devices is growing exponentially and mobile device usage continues to expand, research focused on optimizing such systems for performance and low power consumption across a variety of hardware platforms will be increasingly important.

Real-world Applications and Implementations

Upgrading Secure Communication Protocols: When extended to more complicated communication protocols, like in blockchain and cloud computing systems, applying the hybrid encryption model can greatly increase security. This is especially true in the areas of data integrity and privacy.

Securing IoT Networks: With the spread of IoT devices, protecting secure transmission within these networks is now imperative. In particular, the possibility of using hybrid cryptographic models in IoT networks is full of possibilities, especially given the distinctive problems posed by device heterogeneity and resource constraints.

Addressing Emerging Cybersecurity Threats

1. **Combatting Advanced Cyber Threats**: As cyber threats become more complex, research is needed into hybrid cryptographic models capable of resisting this variety of threat.It involves the analysis of possible weaknesses and countermeasures that are effective against different attack paths.
2. **Educational and Training Initiatives**: Along with these technical developments, there is a need for the nurturing of educational programs to bring forth individuals who are capable not only in using and applying cryptography but also in developing it. This means not only knowing existing cryptographic models, but also being able to develop new ones capable of meeting future security threats.

In short, the direction of cryptographic research is not only to develop stronger encryption methods but also ensure that these become adaptable and efficient enough for increasingly demanding digital environments. Adding hybrid cryptography into the equation, we are ever closer to a future in which digital security is strong and yet flexible enough for today's world.

References

[1] Stallings, W. (2020). *Cryptography and Network Security: Principles and Practice*. Prentice Hall.

[2] Diffie, W., & Hellman, M. (1976). *New Directions in Cryptography*. IEEE Transactions on Information Theory, 22(6), 644-654.

[3] Boneh, D., & Shoup, V. (2020). *A Graduate Course in Applied Cryptography*. Draft version 0.5.

[4] Paar, C., & Pelzl, J. (2010). *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer.

[5] Katz, J., & Lindell, Y. (2020). *Introduction to Modern Cryptography*. Chapman and Hall/CRC.

[6] 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.

[7] National Institute of Standards and Technology (NIST). (2020). *Post-Quantum Cryptography*.

[8] Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of Applied Cryptography*. CRC Press.

[9] Gershenfeld, N., & Chuang, I. L. (1998). *Quantum Computing with Molecules*. Scientific American, 278(6), 66-71.

[10] Nakamoto, S. (2008). *Bitcoin: A Peer-to-Peer Electronic Cash System*.

[11] Z. Wang, S. Wu and Z. Gu, "A RSA algorithm simulation method using C language," 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), Xiamen, China, 2019, pp. 461-464,doi: 10.1109/EITCE47263.2019.9094826.