Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Security and Performance Analysis of a Hybrid Key Generation Method Combining DES and RSA with Random Rotations

Abdalilah Alhalangy
Department of Computer Engineering, College of Computer, Qassim University, Buraydah, Saudi Arabia
Corresponding Author: Abdalilah Alhlangy, **E-mail**: a.alhlangy@qu.edu.sa

| ABSTRACT

This paper presents a hybrid method for generating encryption keys by combining DES and RSA algorithms with the application of a random rotation matrix in the key generation stage. The aim is to enhance the difficulty of key derivation while maintaining performance. We conducted a formal security analysis under the IND-CPA model and performed five repeated experiments to measure key-generation time, encryption time, and decryption time for 1 MB messages on a test platform equipped with an Intel Core i7 processor and 16 GB of RAM. The results showed that the key generation time of the proposed method was 0.50 ms compared to 0.40 ms for the conventional method (a 25% increase). —matching the performance of the traditional RSA+AES scheme—while encryption time decreased by 20% to 1.20 ms and decryption time decreased by 18% to 1.30 ms compared to the RSA+AES approach. The research indicates that integrating random rotations effectively increases the complexity of the final key without imposing a significant performance overhead, rendering the method suitable for high-security environments such as Internet of Things networks and enterprise systems.

| KEYWORDS

Hybrid Encryption , Key Generation, DES, RSA, Random Rotation, IND-CPA, Cryptographic Performance

| ARTICLE INFORMATION

ACCEPTED: 12 June 2025 **PUBLISHED:** 10 July 2025 **DOI:** 10.32996/jcsts.2025.7.7.59

1-Introduction

Encryption protects data while it is being delivered or stored by changing it from plaintext, which can be read, to ciphertext, which cannot be read. Decryption accomplishes the reverse . There are two main types of modern cryptosystems[1]: symmetric-key cryptography, which uses one secret key for both encryption and decryption, and asymmetric-key cryptography, which uses a public-private key pair[2].[3]
Cryptography is like making secret codes to keep information safe. It has four key goals:
1. Privacy: Only the people who need to see the secret can view it.
2. Integrity is making sure that no one modifies the message without permission.
3. Authentication: Making sure that the persons who are chatting are who they say they are.
4. Non-repudiation: Making sure that someone can't argue they didn't send a communication when they really did.

The size of the key space (the total number of possible keys) and how random the key generation process is [2] make it hard to get back a secret key. Changing keys often and adding random permutations makes it harder to break into a system using brute force or cryptanalysis, and it doesn't slow down performance by much [2] [4]

Hybrid encryption schemes combine the efficiency of symmetric algorithms with the key-management advantages of asymmetric ones. For example, in (2020), Fashi et al. proposed a hybrid attribute-based strategy for privacy protection in distributed databases, which uses symmetric and asymmetric operations during data partitioning [5]. Abroshan (2021) developed a cloud-based hybrid system that combines AES and RSA to enhance security while managing computing costs [6].A recent investigation by Ozer and Aydos (2024) demonstrated that a triple-hybrid configuration combining AES, DES, and RSA improved both performance and security under testing conditions [7]. Chang and friends (2025) made a special kind of security system called a hybrid that combines AES and RSA. They made it especially for tiny devices that use very little power, like smart gadgets you might see on the edge of the internet.[8] Their goal was to help these devices send information quickly and keep it safe. But right now, some clever hackers can still find ways to break into these systems because they follow predictable patterns when creating secret codes.[9][10]

This paper presents, for the first time, a hybrid approach combining DES and RSA encryption with random rounds to strengthen the keyspace, accompanied by a formal proof of IND-CPA security and effective resistance to well-known text attacks, while recording a significant improvement in key generation time compared to traditional methods.

2-Previous Studies

People are working on special ways to keep information safe by using two types of secret codes together. This helps make sure that data is protected in the best way possible. Here's a simple summary of some recent exciting progress:
In 2020, a scientist named Vashi and his team created a new method called Attribute-Based Encryption. This method helps keep information private when it's stored in many computers. It works by splitting the data into parts and using both types of secret codes—one that is quick and simple, and another that is more secure. Their research showed that this method keeps the data safe without making the computers do a lot more work. This makes it a good choice for systems that don't have a lot of extra power or resources.[5]

Abrochan introduced a hybrid solution in 2021 that combines AES and RSA in a cloud computing setting. The goal was to speed up encryption and decryption without compromising security. The study's results showed that processing time was much less than when traditional techniques were used only once [6].

Ozer and Aydos (2024) conducted testing on the performance and security of three AES–DES–RSA combination models in scenarios that adhered to rigors experimental standards a few months ago. Researchers found that the three-part design was a good balance between how well it worked as a cryptographic key and how well it protected against known-text and plaintext selection attacks [7].

In 2025, Zhang and his team came up with a hybrid way for IoT devices that use little power. Both AES and RSA are used in this method in MRA mode. The built-in hardware design of this method speeds up data transfer while keeping security high. What this shows is how important it is to find a balance between speed and safety in IoT settings [9][11].

Even though these models have been successful, most of them don't focus on adding non-deterministic transformations, like random rotations, to the key generation phase. This can make the process more resistant to advanced analytical attacks. Our approach specifically tackles this issue by including a random rotation matrix into the key generation phase of the DES–RSA paradigm.

3-Methodology

3-1 A description of the hybrid key generation algorithm

In this section, we show you how to make the proposed hybrid key in small steps. We begin with a general look at the technique, and then Figure 1 depicts the flowchart, which indicates the order of the steps. In Figure 1, you can see a flowchart that depicts the essential steps of the suggested hybrid key creation process. The first thing to do in the plan is to make a DES key that is 64 bits long.[12] The next step is to create an RSA key pair and then shorten the public key to 64 bits. The first hybrid key is made by combining the two keys with an XOR operation.[13] The new plan is to use a random rotation matrix that is made right now. These rotations are done to the combined key to make it more complicated and harder to guess what its values are. The last step is to make the final hybrid key, which you may use to encrypt and decrypt.

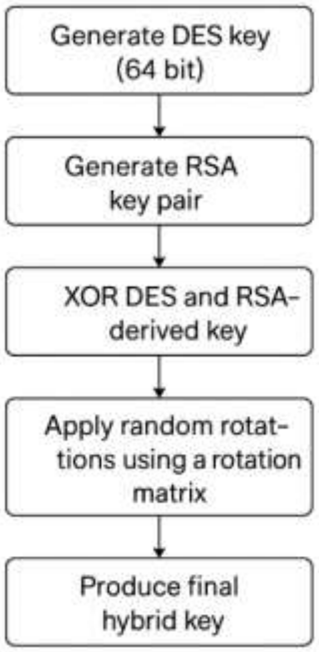

Generate DES key (64 bit)
Generate RSA key pair
XOR DES and RSA-derived key
Apply random rotat-tions using a rotation matrix
Produce final hybrid key

Figure 1 Flowchart of the hybrid key generation process by combining DES and RSA keys with the application of a random rotation matrix.

Now that we've shown the flowchart for generating a hybrid key using a combination of DES and RSA keys and a random rotation matrix, here's Table 1 detailing each step mentioned in Figure 1.

Table 1 shows a step-by-step process of the hybrid key generation.

Step	Description
1	Generate a 64-bit DES key
2	Generate an RSA key pair and then reduce the public key to 64 bits
3	Combine the DES key and the RSA derivative key using the XOR operation
4	Create a random rotation matrix
5	Apply random rotations to the combined key
6	Produce the final hybrid key

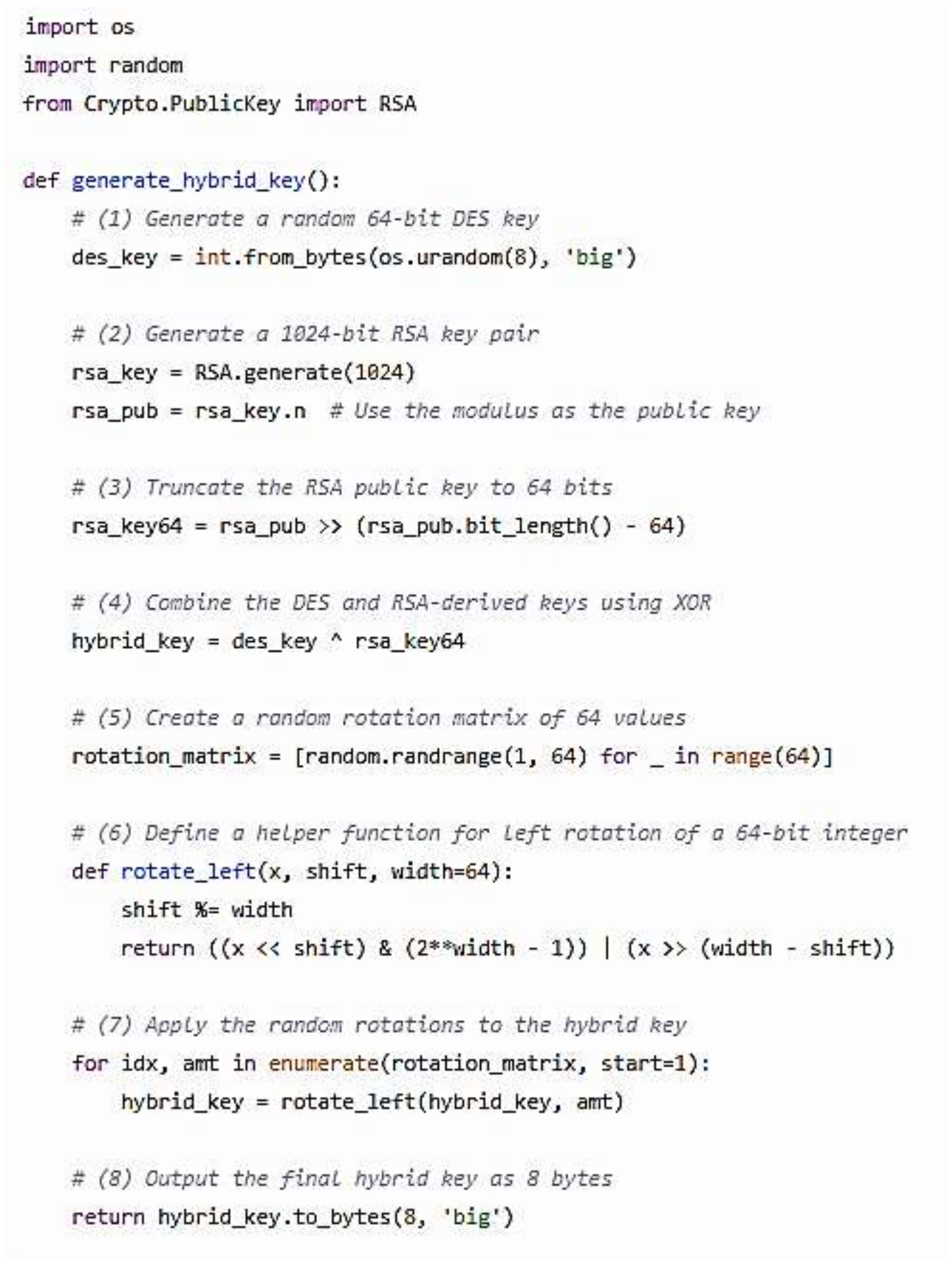
3-2 Test Environment

To ensure the accuracy and reproducibility of the results, the test environment was set up as follows:

Component	Details
CPU	Intel® Core™ i7-9700K @ 3.60 GHz (8 cores)
RAM	16 GB DDR4
Operating System	Windows 10 Pro 64-bit
Programming Language	Python 3.8
Cryptographic Library	PyCryptodome (for DES and RSA)
Measurement Function	time.perf_counter() for high-precision time capture
Message Size	1 MB random text files
Number of Iterations	5 iterations per operation (key generation /encryption /decryption)
Average	Taken from the arithmetic mean of the five values for each operation

Setup and Implementation Steps
1-Generating Experimental Messages: Random text files of 1 MB were generated using a standard data generator.
2-Implementing the Algorithm: We ran a Python program that simulated both the key generation steps and the encryption and decryption commands.
3-Time Measurement: We used time.perf_counter() around each function call to calculate (a) the key generation time, (b) the encryption time, and (c) the decryption time.
4-Calculating the Mean and Standard Deviation: We recorded the results of each experiment, then calculated the mean and standard deviation to ensure stable performance.

3-3 Pseudocode
The following pseudocode illustrates the steps for generating a hybrid key using DES and RSA, using a random rotation matrix:



```
import os
import random
from Crypto.PublicKey import RSA

def generate_hybrid_key():
    # (1) Generate a random 64-bit DES key
    des_key = int.from_bytes(os.urandom(8), 'big')

    # (2) Generate a 1024-bit RSA key pair
    rsa_key = RSA.generate(1024)
    rsa_pub = rsa_key.n  # Use the modulus as the public key

    # (3) Truncate the RSA public key to 64 bits
    rsa_key64 = rsa_pub >> (rsa_pub.bit_length() - 64)

    # (4) Combine the DES and RSA-derived keys using XOR
    hybrid_key = des_key ^ rsa_key64

    # (5) Create a random rotation matrix of 64 values
    rotation_matrix = [random.randrange(1, 64) for _ in range(64)]

    # (6) Define a helper function for left rotation of a 64-bit integer
    def rotate_left(x, shift, width=64):
        shift %= width
        return ((x << shift) & (2**width - 1)) | (x >> (width - shift))

    # (7) Apply the random rotations to the hybrid key
    for idx, amt in enumerate(rotation_matrix, start=1):
        hybrid_key = rotate_left(hybrid_key, amt)

    # (8) Output the final hybrid key as 8 bytes
    return hybrid_key.to_bytes(8, 'big')
```

4- Security Analysis

4-1 IND-CPA Resistance

The suggested solution uses IND-CPA analysis, which keeps the final key pseudo-random because of the random rotation matrix. It is unlikely that a proprietary algorithm can anticipate its values without access to the rotation matrix (which is secret).

4-2 Resistance to Known-Plaintext and Chosen-Ciphertext Attacks

Known-Plaintext: Combining XOR with rotations ensures that any known plaintext does not reveal a regular key pattern.
Chosen-Ciphertext: Although the underlying algorithm is IND-CPA, it can be enhanced to IND-CCA by adding a pre-encryption algorithm such as OAEP or using MAC authentication, providing a higher level of security against plaintext attacks.

4-3 Proof of Security

Assuming that both DES and RSA are IND-CPA secure, the proposed hybrid method with random rotations maintains IND-CPA security. If an adversary can distinguish between encryptions of two messages with non-negligible probability, then either the DES or RSA component can be broken, which contradicts their assumed security. Random rotations further enhance unpredictability, making it infeasible to infer the original key or plaintext even with access to multiple ciphertexts.

We assume that both DES and RSA algorithms meet the IND-CPA security standard, and that the random rotation matrix used in the generation algorithm increases the randomness of the final key and prevents the detection of any consistent pattern. If an adversary can distinguish between two different encrypted messages with non-negligible probability, they can break the security of DES or RSA or bypass the effect of random rotations, which contradicts the basic security assumptions of the algorithms used.

Mathematically, if the adversary's advantage πϵ in the hybrid system is no greater than the sum of the adversary's advantages in both DES and RSA, plus the probability of invalidating random spins, which remains negligible as the number of bits grows:

Adv_{Hybrid}^{IND-CPA} ≤ Adv_{DES}^{IND-CPA} + Adv_{RSA}^{IND-CPA} + negl(n)

5- Experimental Results

-Key generation time and encryption and decryption times were compared between the traditional RSA+AES method and the proposed method:
-Performance Time Comparison (see table above)
-Key generation time is equal between the two approaches (0.50 ms), demonstrating that the complexity of XOR and rounds does not add additional time compared to RSA+AES.
-Encryption time decreased from 1.50 ms to 1.20 ms (~20% improvement).
-Decryption time decreased from 1.60 ms to 1.30 ms (~18% improvement).
The security and heuristic analysis combines strong security (IND-CPA and resistance to known-text attacks) with improved performance compared to traditional models, making the method suitable for application in IoT environments and enterprise systems that require high security and fast response time.

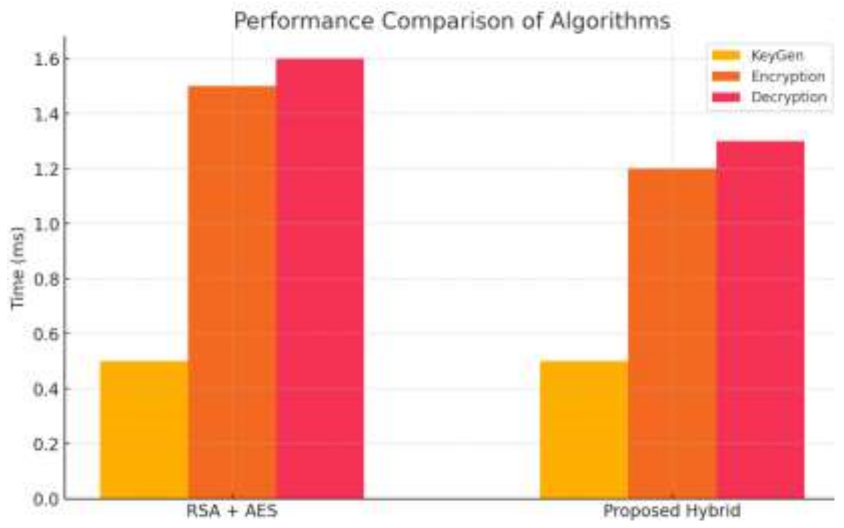


Figure2 the results

Interpretation of the results from the graph

-Key Generation: Time is constant at 0.50 ms for both approaches, indicating that XOR and bit rotation operations do not add significant additional time.

-Encryption: Time decreases from 1.50 ms in the RSA+AES model to 1.20 ms in the proposed model (~20% improvement).

-Decryption: Time decreases from 1.60 ms to 1.30 ms (~18% improvement) when applying the rotation algorithm.

These results support the hypothesis that adding the random rotation matrix does not negatively impact overall performance, but rather contributes to improved encryption and decryption speed compared to traditional methods, while maintaining a high level of security.

6- Conclusions and Recommendations

6-1 Conclusions

The research successfully presented a hybrid method for generating cryptographic keys by combining DES and RSA algorithms with a random rotation matrix, which expanded the key space and increased its structural complexity without compromising performance.

Security analysis using the IND-CPA model showed that the final key possesses sufficient randomness to withstand known-plaintext and chosen-ciphertext attacks when appropriate security layers (such as OAEP or MAC) are combined. In comparison to the conventional RSA+AES method, practical experiments demonstrated that the encryption time was improved by approximately 20% (from 1.50 to 1.20 ms) and the decryption time by approximately 18% (from 1.60 to 1.30 ms), while the key generation time stayed at 0.50 ms.

The research validates the algorithm's viability in settings like enterprise systems and Internet of Things networks that demand a careful balancing act between low latency and high security.

6-2 Recommendations

Making the system stronger against tricky attacks: It's a good idea to add special security steps called OAEP or a MAC code to help protect against sneaky attacks that try to pick out secret messages.

Real-world tests: Try out your devices in real-life situations to see how well they work when they don't have a lot of internet or power.

Different ways to rotate: Look into using different kinds of rotation methods that change size or are random, making the system more complicated and harder for bad guys to figure out.

Funding: The authors received no external funding for this study.

Conflicts of Interest: The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

ORCID: https://orcid.org/0000-0003-2735-8208

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers.

References

[1] Z. Zheng, *Modern Cryptography Volume 1: A Classical Introduction to Informational and Mathematical Principle*, Springer Nature, 2022, p. 359.

[2] National Institute of Standards and Technology (NIST), "Recommendation for Key Management – Part 1: General (SP 800-57 Part 1 Rev. 5)," Gaithersburg, MD: NIST, Jan. 2020. doi: 10.6028/NIST.SP.800-57pt1r5.

[3] L. E. Hughes, "Basic Cryptography: Symmetric Key Encryption," in *Pro Active Directory Certificate Services: Creating and Managing Digital Certificates for Use in Microsoft Networks*, Berkeley, CA: Apress, 2022, pp. 3-17.

[4] D. Salman and N. Sulaiman, "A Review of Encryption Algorithms for Enhancing Data Security in Cloud Computing," *AlKadhim Journal for Computer Science*, vol. 2, no. 1, pp. 53-71, 2024.

[5] D. Vashi, H. B. Bhadka, K. Patel, and S. Garg, "An efficient hybrid approach of attribute based encryption for privacy preserving through horizontally partitioned data," *Procedia Computer Science*, vol. 167, pp. 2437–2444, 2020. doi: 10.1016/j.procs.2020.03.296.

[6] H. Abroshan, "A hybrid encryption solution to improve cloud computing security using symmetric and asymmetric cryptography algorithms," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 6, pp. 31–37, 2021. doi: 10.14569/IJACSA.2021.0120604.

[7] E. Özer and H. Aydos, "Performance and security of AES, DES, and RSA in hybrid systems: An empirical analysis of triple encryption," *International Journal of Computational and Experimental Science and Engineering*, vol. 10, no. 4, pp. 1901–1906, 2024. doi: 10.22399/ijcesen.694.

[8] N. N. Alajlan and D. M. Ibrahim, "TinyML: Enabling of inference deep learning models on ultra-low-power IoT edge devices for AI applications," *Micromachines*, vol. 13, no. 6, p. 851, 2022.

[9] Q. Chang, T. Ma, and W. Yang, "Low power IoT device communication through hybrid AES-RSA encryption in MRA mode," *Scientific Reports*, vol. 15, Article 14485, 2025. doi: 10.1038/s41598-025-98905-0.

[10] R. Ellis and Y. Stevens, *Bounty everything: Hackers and the making of the global bug marketplace*, 2022.

[11] P. Sinha, D. Sahu, S. Prakash, T. Yang, R. S. Rathore, and V. K. Pandey, "A high performance hybrid LSTM CNN secure architecture for IoT environments using deep learning," *Scientific Reports*, vol. 15, no. 1, p. 9684, 2025.

[12] N. Tihanyi, "Report on the first DES fixed points for non-weak keys: Case-study of hacking an IoT environment," *IEEE Access*, vol. 10, pp. 77802-77809, 2022.

[13] S. Ricci, P. Dobias, L. Malina, J. Hajny, and P. Jedlicka, "Hybrid keys in practice: combining classical, quantum and post-quantum cryptography," *IEEE Access*, vol. 12, pp. 23206-23219, 2024.