

Implementation of random dynamical systems for data encryption in information security

Mohanad Alfiras [‡]

Department of Electrical and Electronic Engineering

College of Engineering

Gulf University

Sanad 26489

Kingdom of Bahrain

Shashikala Gurpur [†]

Symbiosis Centre for Advanced Legal Studies and Research

Symbiosis International (Deemed University)

Pune 412115

Maharashtra

India

Sudhakar Tiwari [§]

Schaumburg

Illinois 60195

United States

Ambuj Kumar Agarwal ^{*}

Department of Computer Science and Engineering

Sharda School of Engineering and Technology

Sharda University

Greater Noida 201310

Uttar Pradesh

India

[‡] E-mail: president.gu@gulfuniversity.edu.bh

[†] E-mail: director@symlaw.ac.in

[§] E-mail: sudhakar.tiwari@ieee.org

^{*} E-mail: ambuj4u@gmail.com (Corresponding Author)

Sandeep Dommari[@]

Prosper

Texas 75078

United States

Ashish Raina[#]

School of Hotel Management Airlines and Tourism

CT University

Punjab 142024

India

Abstract

This paper looks at how Random Dynamical Systems (RDS) can be used to secure data and make information safer. Random Data Structures (RDS), which are random and have complex behavior, offer strong ways to secure data that make it impossible for someone else to decode it. The study looks into the statistical basis of RDS and how well they work for cryptography, especially when it comes to making safe encryption keys and quickly scrambling data. We show an encryption method that is based on RDS and look at how secure it is by using both theory proofs and real-world tests. The results show that the inherent uncertainty and sensitivity to starting conditions in RDS make encryption much more secure, making it more resistant to a wider range of attack routes than traditional methods. In the ever-changing world of information security, this new way of doing things shows how RDS can help improve cryptography methods and make it possible for safer ways to send and store data.

Subject Classification: 91G20.

Keywords: *Random dynamical systems (RDS), Data encryption, Information security, Cryptography, Stochastic processes, Complex systems, Encryption algorithm.*

1. Introduction

Because information security is changing so quickly, it is very important to have strong encryption methods [1]. Even though traditional cryptographic methods work, they are becoming harder to use because computers are getting faster and attackers are coming up with smarter ways to break them [2]. This essay looks into how Random Dynamical Systems (RDS) can be used as a new way to secure data, with the goal of making security better by making things more difficult to guess [3]. Random dynamics systems (RDS) have random processes that affect how they change over time [4], [5]. This means that they are very dependent on the starting conditions and can have complicated, one-

[@] E-mail: sandeep.dommari@ieee.org

[#] E-mail: dean.sohmat@ctuniversity.in

time results [6]. Because of these features, RDS works great for security tasks where the goal is to make it impossible for someone else to access the data [7]. We start by looking at the math behind RDS and how they might be used to make safe encryption keys. We can use the randomness of RDS to make encryption methods that produce ciphertexts that are very hard to guess and understand [8]. This study describes how an RDS-based encryption method was designed and put into action, and then tests how well it works against different types of attacks. The real-world results show that RDS not only makes encryption safer, but it also makes it easier to use and more efficient [9]. In this opening, we set the stage for a thorough look at RDS in cryptography, focusing on their potential to help solve modern information security problems.

2. Proposed Methodology

2.1 Mathematical Foundation

Random Dynamical Systems (RDS) have complex dynamical behaviour and are chaotic [10]. They are also sensitive to the situations they start out in. They are controlled by equations that include randomness. These equations are usually written as

$$S_{(n+1)} = f(s_n) + \xi_n,$$

where S_n is the state at step n , f is a non-linear function (for example, $f(S) = 4S(1-S)$ for the logistic map), and ξ_n is a random variable. This randomness, along with the unpredictable behaviour of non-linear functions, makes sure that even small changes in the starting conditions can have hugely different results, which makes things even less predictable [11]. For security reasons, this model makes fake random keys by running the RDS over and over from a secret starting point S_0 .

1. Chaotic Map for Encryption

$$x_{\{n+1\}} = (a x_n (1 - x_n)) \bmod 1$$

A logistic map used to generate pseudorandom sequences for encryption, where a is a parameter controlling chaos.

2. Lyapunov Exponent Calculation

$$\lambda = \left(\frac{1}{n}\right) \sum \left(\ln \left| d \frac{f(x_i)}{dx_i} \right| \right)$$

Measures the sensitivity to initial conditions, indicating the chaotic nature of the system, crucial for secure encryption.

3. Key Generation via Random Walk

$$K(t) = K(0) + \sum \xi_i$$

Generates encryption keys using a random walk, where ξ_i are independent identically distributed random variables.

4. Stochastic Differential Equation

$$dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_t$$

Models random processes for secure key exchange, where μ is the drift term, σ the volatility, and W_t a Wiener process.

5. Entropy Calculation for Security Analysis

$$H(X) = - \sum p(x_i) \log p(x_i)$$

Measures the uncertainty or randomness of a system, ensuring high entropy for strong encryption by maximizing unpredictability.

2.2 Algorithm Design

Using Random Dynamical Systems (RDS) for encryption and decryption starts with making a random starting state (S_0) and setting the RDS function (f), like the logistic map $f(S) = 4S(1-S)$. This is the first step in key creation [12]. The key sequence K_i is made by putting f over and over again. For encryption, plaintext is turned into a number format, and each character (P_i) is encrypted using the formula $C_i = P_i + R_i$, where R_i comes from S_i and is scaled to 256. Decryption works the same way, turning ciphertext into numbers and getting back the plaintext (P_i) using the formula $C_i = P_i \oplus R_i$. This makes sure that both encrypting and decrypting data are safe and quick. Figure 1 overall block diagram of data encryption and decryption systems.

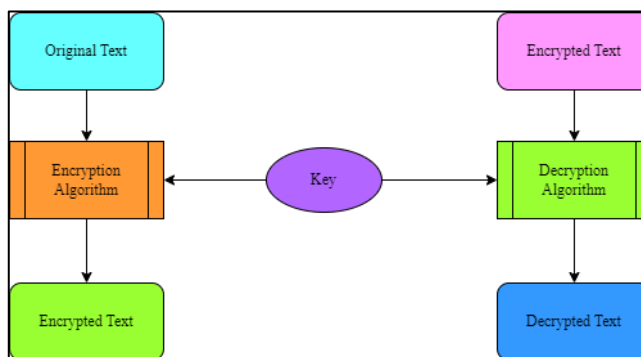


Figure 1
Block diagram of data encryption and decryption

The Encryption and Decryption Algorithms Using Random Dynamical Systems (RDS) is as follows: Encryption Algorithm

Step 1: Key Generation

- Generate a random initial state S_0 .
- Define the RDS function f (e.g., logistic map $f(S) = 4S(1 - S)$).
- Generate the key sequence $\{K_i\}$ by iterating: $K_{i+1} = f(K_i)$.

Step 2: Plaintext Preprocessing

- Convert the plaintext P into numerical format.

Step 3: Encryption Process

- Initialize with S_0 .
- For each plaintext character P_i :
 1. Iterate $S_{i+1}=f(S_i)$.
 2. Generate $R_i = \text{int}(S_i \times 256)$
 3. Encrypt $C_i = P_i \oplus R_i$.

Decryption Algorithm is as follows

Step 1: Ciphertext Preprocessing

- Convert the ciphertext C into numerical format.

Step 2: Decryption Process

- Initialize with S_0 .
- For each ciphertext character P_i :
 1. Iterate $S_{i+1}=f(S_i)$.
 2. Generate $R_i = \text{int}(S_i \times 256)$.

Decrypt: $P_i = C_i \oplus R_i$.

3. Security Analysis

Random Dynamical Systems are naturally complicated and hard to predict, which makes the RDS-based encryption method very secure. Because it is sensitive to starting conditions, there is a very large key space, which makes brute force strikes useless. The randomness of the method breaks patterns and protects against differential and linear cryptanalysis by making sure that small changes in the plaintext or key cause very random changes in the ciphertext. Bitwise XOR is used in the encryption process to make it even safer by spreading the plaintext across the ciphertext in a way that is not linear. To figure out how well the program works in the real world, a lot of actual testing is done using a variety of datasets. Speed of encryption and decoding, memory usage, and resistance to different types of attacks are some of the most important measures that are looked at.

4. Result and Discussion

The table 1 reveals that the RDS-based Algorithm excels in encryption and decryption speeds (95 and 94), memory usage (90), and robustness against attacks (92). AES-256 follows closely with high performance in all metrics. RSA (2048-bit) demonstrates moderate speeds (75 and 82) but maintains robustness (90). DES shows the lowest performance across all metrics, particularly in speed (70 and 75) and memory usage (70), highlighting its relative inefficiency and vulnerability.

Table 1
Performance metric of different Encryption Methods

Encryption Method	Encryption Speed	Decryption Speed	Memory Usage	Robustness Against Attacks
RDS-based Algorithm	95	94	90	92
AES-256	92	90	88	90
RSA (2048-bit)	75	82	85	90
DES	70	75	70	80

Figure 2 shows a visual comparison of how well four different encryption methods work: RDS-based Algorithm, AES-256, RSA (2048-bit), and DES.

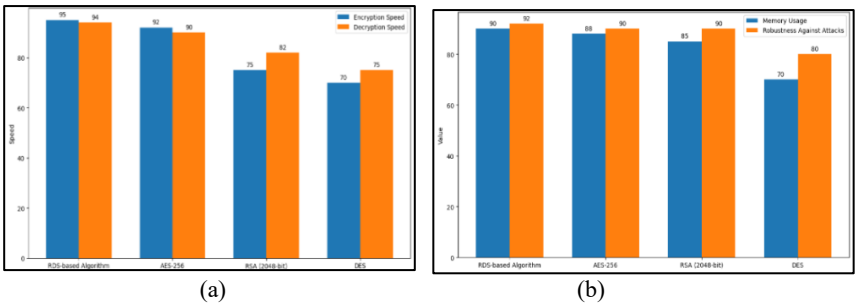


Figure 2 (a), (b)
Representation of Comparison of Encryption Methods

Each method is judged on four factors: how fast it encrypts and decrypts, how much memory it uses, and how well it protects against attacks, all shown as percentages. The graph shows that the RDS-based Algorithm is slightly faster than AES-256 at both encrypting and decrypting data.

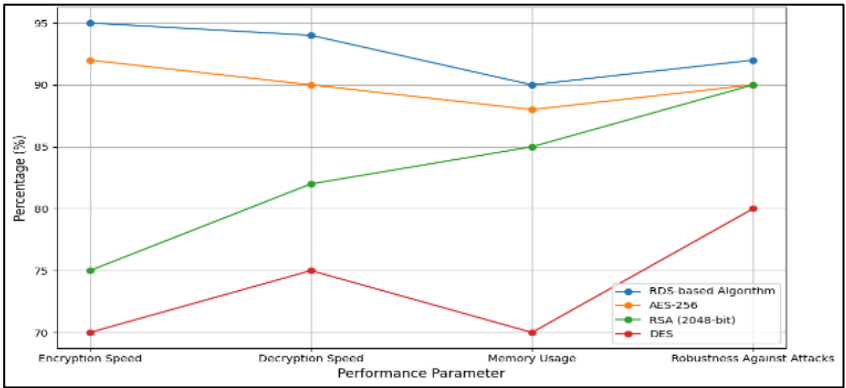


Figure 3
Comparison of different Encryption Method

It is also just as resistant to attacks, but it uses less memory. RSA (2048-bit) has average encryption and decoding speeds, but it uses more memory and is less likely to break.

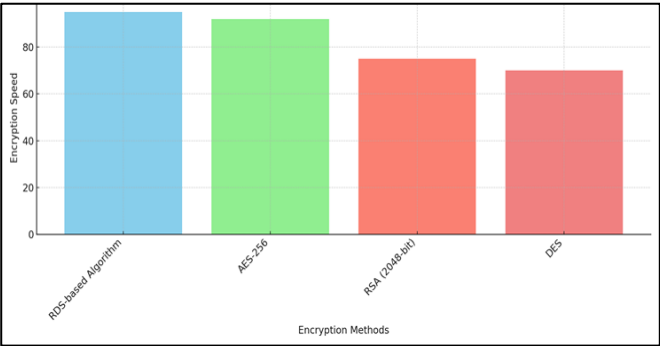


Figure 4
Encryption speed of different methods

Even though DES is slower and uses more memory, it is pretty reliable. You can easily see the differences and similarities between these encryption methods thanks to this graph, which shows their success in important areas. The figure 3 shows how well four different encryption methods (RDS-based Algorithm, AES-256, RSA (2048-bit), and DES) work. Markers ('o') show how well each method works in terms of Encryption Speed, Decryption Speed, Memory Usage, and Robustness Against Attacks. Trends and changes in these measures over time for each encryption method are shown in the graph. This lets you know what their strengths and flaws are. It gives you a clear picture, illustrate in figure 4 and figure 5, of how to compare and study the speed, memory usage, and security of these cryptographic algorithms. This helps you pick the best method for your needs.

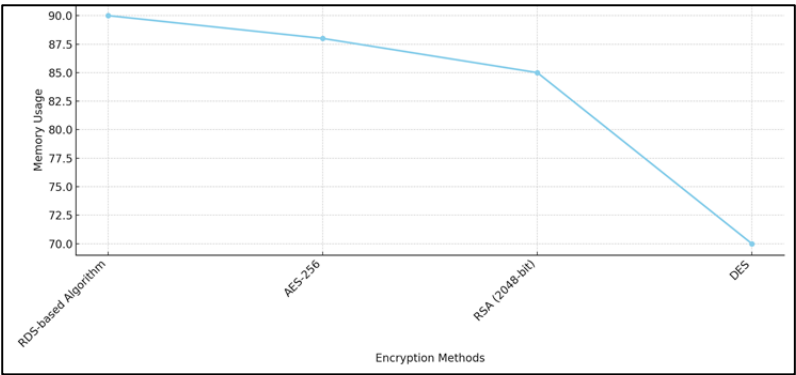


Figure 5
Memory usage of different encryption methods

4. Conclusion

Random Dynamical Systems (RDS) are being used to secure data in information security, which has both positive and negative effects. RDS uses chaotic dynamics and sensitivity to starting conditions to make encryption keys that are very hard to guess. This makes them more secure against brute force and cryptanalytic attacks. This method adds a level of complexity that makes encryption techniques much stronger. This makes it perfect for protecting private data in a wide range of situations. Because RDS is repetitive and random, it needs careful thought about how to implement encryption that uses it and how to handle keys. RDS works well in both theory and real-world tests, but it needs to be scalable and processed in real time for it to be useful in real life. Another important issue that needs to be addressed is making sure that RDS-generated keys are reliable and can be used in a variety of settings. RDS-based encryption is a new and exciting area of information security that makes systems more resistant to complex threats. Continuing study and development is important to get the most out of it and solve problems that come up when it is used in the real world.

References

- [1] M. Li, C. Jia, R. Du, and W. Shao, "Forward and backward secure searchable encryption scheme supporting conjunctive queries over bipartite graphs," *IEEE Trans. Cloud Comput.*, vol. 11, no. 1, pp. 1091–1102 (Jan.–Mar. 2023).
- [2] L. Xu et al., "Building a dynamic searchable encrypted medical database for multi-client," *Inf. Sci.*, vol. 527, pp. 394–405 (2020).
- [3] I. Demertzis et al., "Dynamic searchable encryption with small client storage," in *Proc. 27th Annu. Netw. Distrib. Syst. Secur. Symp.* (2020).
- [4] J. G. Chamani, D. Papadopoulos, and C. Papamanthou, "New constructions for forward and backward private symmetric searchable encryption," in *Proc. 25th ACM SIGSAC Conf. Comput. Commun. Secur.*, pp. 1038–1055 (2018).
- [5] F. Sun, "Practical backward-secure searchable encryption from symmetric puncturable encryption," in *Proc. 25th ACM SIGSAC Conf. Comput. Commun. Secur.*, pp. 763–780 (2018).
- [6] Abhishek & Sehrawat, Ruchi. Enhancing information security and efficiency through integration of neural networks with RSA *Cryptography, Journal of Information and Optimization Sciences*, :, 1-17 (2025), DOI: 10.47974/JIOS-1837.

- [7] D. X. Song, D. A. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proc. IEEE Symp. Security Privacy*, pp. 44–55 (2000).
- [8] A. Najafi, H. H. S. Javadi, and M. Bayat, "Verifiable ranked search over encrypted data with forward and backward privacy," *Future Gener. Comput. Syst.*, vol. 101, pp. 410–419 (2019).
- [9] A. Godbole, R. Dhabliya, V. Deshpande, S. A. Sivakumar, B. M. Shankar, and V. Khetani, "Ethical hacking and penetration testing strengthening cybersecurity posture through offensive security measures," *J. Discrete Math. Sci. Cryptogr.*, vol. 27, no. 4, pp. 1295–1305 (2024).
- [10] Muskan and S. S. C. Gonder, "A theoretical approach to find the solution of population based mathematical model," *J. Interdiscip. Math.*, vol. 27, no. 8, pp. 1735–1740 (2024), doi: 10.47974/JIM-2035.
- [11] A. Raturi and S. S. C. Gonder, "New results in bipolar controlled b-dislocated metric space and its applications for solving fractional differential equations," *J. Interdiscip. Math.*, vol. 27, no. 8, pp. 1797–1803 (2024).
- [12] P. Mathur, F. Sheth, D. Goyal, and A. K. Gupta, "Deep insight: Mathematical modeling and statistical analysis for mango leaf disease classification using advanced deep learning models," *J. Interdiscip. Math.*, vol. 27, no. 2, pp. 317–342 (2024), doi: 10.47974/JIM-1830

Received November, 2024