University Of AL-Qadisiyah

Available online at www.qu.edu.iq/journalcm

JOURNAL OF AL-QADISIYAH FOR COMPUTER SCIENCE AND MATHEMATICS

ISSN:2521-3504(online) ISSN:2074-0204(print)

Performance Analysis for RC6 and PRESENT Encryption Algorithms in Cloud Environment

Doaa S. Salman[a]*, Huda H. Ali[b]

[a]Imam Alkadhim College, Baghdad 10061, Iraq.Email: doaa.sa.salman@gmail.com

[b]Imam Alkadhim College, Baghdad 10087, Iraq.Email: hudahamdan@iku.edu.iq

ARTICLE INFO

Article history:
Received: 08 /09/2024
Rrevised form: 19 /09/2024
Accepted : 1 /10/2024
Available online: 30 /12/2024

Keywords: NIST Compliance, Lightweight Algorithms, Cloud Environment, RC6 Algorithm, PRESENT Algorithm.

Each keyword to start on a new line

ABSTRACT

Organizations and individuals increasingly rely on cloud services for data storage and transmission, and the need for robust and efficient encryption algorithms is more critical than ever. This paper rigorously examines the performance and NIST compliance of two lightweight encryption algorithms, RC6 and PRESENT, within cloud computing contexts. The paper assesses the algorithms' performance metrics, such as encryption time, decryption time, space-time, memory used, CPU consumption, avalanche effect, entropy, and energy consumption, in addition to their security features. Also, utilizing fifteen NIST SP 800-22 cryptographic tests to ensure secure and efficient practices, including recent security analyses to assess the two algorithms' behavior, both algorithms present a distinct array of compromises between effectiveness and protection.

The results function as a strong regulator for the decision-making process in cloud security, offering advantages to both professionals in the field and experts in cryptography, it shows RC6 outperforms PRESENT in encryption (5.334s vs. 6.937s), decryption (6.035s vs. 6.191s), and space-time efficiency (1189 bytes/s vs. 41 bytes/s). While PRESENT uses less memory after encryption (41.27 MB vs. 67.63 MB), RC6 has slightly lower CPU consumption (4.50% vs. 4.90% for encryption) and higher energy efficiency (0.901J vs. 0.956J). Deep analysis is implemented to balance security and efficiency, and after implementing the two algorithms to encrypt the same file the result of the analysis was, that RC6 stands out as a particularly strong contender. It offers robust security features and is more efficient in terms of computational time, resource utilization, and energy consumption. PRESENT, while secure, tends to consume more resources and takes longer for encryption and decryption. However, its simpler design might offer some advantages in certain scenarios, though this comes with trade-offs in performance and resource efficiency.

MSC..

https://doi.org/10.29304/jqcsm.2024.16.41775

1. Introduction

Digital information and cloud computing is common in last years, the protection of sensitive data occurs as a central focus. Encryption serves not only to shield data from unauthorized entry but also to ensure the confidentiality and

∗Corresponding author

Email addresses:

Communicated by 'sub etitor'

integrity of data both in transportation and in saving state. Therefore, the selection of encryption algorithms that strike the suitable balance between security and efficiency is overriding [1].

Lightweight cryptographic algorithms have expanded increase gratitude because of their value and insignificant computational source difficulties, execution them particularly appropriate for cloud-based environments where source efficiency and promptness are crucial [2].

Assessing the efficacy of specific lightweight encryption algorithms in cloud computing settings, ensuring commitment to the stringent criteria set forth by the National Institute of Standards and Technology (NIST). The PRESENT and RC6 algorithms embody a diverse range of encryption methodologies, each possessing distinct advantages and attributes as evaluated by NIST [3].

Furthermore, adherence to recognized security protocols is rigid in the contemporary interconnected global environment. The National Institute of Standards and Technology (NIST) has supplied thorough instructions and suggestions as to encryption and key control, as realized in publications such as NIST Special Publications 800-22 and 800-53. It is essential to ascertain that encryption methods comply with these protocols to maintain strong data security [4],[5]. There are numerous studies that contract with encryption algorithm analysis performance; A study by Omolara et al (2019) observes cloud encryption techniques to control their security for cloud-based applications, aiming to provide guidance for professionals in selecting suitable encryption techniques to optimize functionality while maintaining security measures [6]. Elaine Barker et al (2020), explain upon the import of NIST Special Publication 800-133 Revision 2 in providing counsel on the generation of cryptographic keys. Especially, the publication defines optimal methodologies to producing secure cryptographic keys, whereby the emphasis is laid on randomness, algorithmic procedures, and entropy sources. Similarly, the publication covers commendations for key generation processes, algorithms, and key lengths to fortify the security of cryptographic systems. Hence, the publication stands as a valued resource for items seeking to begin robust cryptographic key handling practices [7].

Panagiotis Podimatas et al. (2022), explore the efficacy of lightweight block ciphers in highly controlled environments. Specifically, it explores into the saturnine family of lightweight ciphers, which have been under consideration as a probable standardization candidate in the ongoing NIST competition. Through this inquiry, it becomes apparent that lightweight cryptography, overall situations that do not necessitate the use of a consistent algorithm, offers several choices for choosing a suitable cipher on an ad hoc basis based on precise needs. This method is adequate for providing practical security solutions in guarded environments [8].

Martin Herman et al (2023), The NIST Cloud Computing Forensic Science Reference Architecture, also mentioned to as NIST SP 800-201, offerings a complete framework for the execution of digital forensic investigations in cloud computing environments. They explains essential perceptions, components, and processes that are integral to cloud forensic analysis, thereby facilitating the navigation of the unique challenges that rise in the context of cloud-based evidence collection and analysis. A spirited focus is the protection of digital confirmation integrity and chain of custody within cloud environments. Consequently, it works as a vital resource in the realm of cloud computing environments [9].

This paper compared two algorithms' performance is powerful as it contributes to knowledge by highlighting their strengths, weaknesses, and applicability, guiding practitioners in making informed choices. It also sets benchmarks for future research and optimization in specific domains, especially in emerging technologies.

The rest of the paper is organized as follows: Section 2 represents the methodology of applying the two algorithms in a specific environment. Section 3 represents the encryption algorithms. Section 4 represents the comparison factors and evaluation methods. Section 5 represents the test NIST Compliance. The experimental result is represented in Section 6, these results have been discussed in Section 7, and Section 8 represents the conclusion.

2. Methodology

The proposed methodology aims to tests which encryption algorithm is more suitable for providing security to data on the cloud environment, when users try to access the data from anywhere and any device. The methodology includes three phases; The first phase is to build a database with many sizes on a local server, such as the XAMPP. The second phase is to execute the encryption algorithms in Python by encrypt the same file of size 112 KB using the two algorithms, to evaluate it using ten criteria. The NIST tests are utilized during the third phase to confirm the randomness and uniform distribution of cryptographic results, which are crucial for the maintenance of data confidentiality.

To add advanced redundancy the key from the chaotic Lorenz map is used; that is a three-dimensional chaotic system showcasing the renowned "butterfly" attractor, which holds significant importance in the fields of weather forecasting and atmospheric simulation [10]. as present equation (1):

$$
\begin{aligned}
&dx/dt = \sigma * (y - x),\\
&dy/dt = x * (\rho - z) - y, \qquad (1)\\
&dz/dt = x * y - \beta * z
\end{aligned}
$$

The Lorenz system consists of three nonlinear differential equations:

- dx/dt = σ * (y - x): This equation models the rate of change of x, where σ\sigmaσ controls the convection rate, and (y−x) represents the horizontal temperature difference between fluid layers.
- dy/dt = x * (ρ - z) - y: This equation represents the rate of change of y, where ρ\rhoρ is the Rayleigh number, driving convection. It captures how vertical temperature differences and resistance impact fluid flow.
- dz/dt = x * y - β * z: This equation describes the rate of change of z, with x*y introducing nonlinearity, and βz\beta representing energy dissipation.

Overall, the Lorenz system models fluid convection and is highly sensitive to initial conditions, making it ideal for generating chaotic keys in encryption due to its unpredictable behavior.

3. Encryption Algorithms

Encryption is a crucial component in guaranteeing the confidentiality, integrity, and authenticity of information across diverse domains, such as communication, finance, healthcare, and cybersecurity [11]. Two of the important encryption algorithms (PRESENT and RC6).

3.1 The PRESENT

Is an ultra-lightweight block cipher designed for resource-constrained devices. It operates on small 64-bit blocks and supports key sizes of 80 or 128 bits. Extremely it is efficient in terms of hardware implementation, making it suitable for IoT devices. In addition, it has a strong resistance against known cryptographic attacks but its Limited block size may not be ideal for some applications, and Key agility (changing keys frequently) may be challenging [12]. The following pseudo-code for the procedure of the Present encryption algorithm.

Algorithm (3.1) expresses the pseudo-code of the Present algorithm:

Algorithm (3.1) PRESENT (state, round Keys, SBox , PBox)

```
Input:
  state: 64-bit integer representing the current state (plaintext)
  round Keys: Array of 64-bit round keys
  SBox: Array of 16 elements representing the S-box
  PBox: Array of 64 elements representing the permutation table
Output:
   state: Updated state after applying the round function (ciphertext)
Begin:
// Initial Key Mixing
Step 1: state ← state XOR round Keys[0]
// Main Loop (Repeating for 31 rounds)
Step 2: for round ← 1 to 31:
 // S-Box Layer
Step 3: for nibble ← 0 to 15:
        nibble value ← (state >> (4 * nibble)) AND 0xF
        substituted value ← SBox[nibble value]
        state ← (state AND ~(0xF << (4 * nibble))) OR (substituted value << (4 * nibble))
// Permutation Layer
Step 4: permuted state ← 0
Step 5: for bit_pos ← 0 to 63:
      bit ← (state >> bit_pos) AND 1
      permuted state ← permuted state OR (bit << PBox[bit_pos])
      state ← permuted state
// AddRoundKey
Step 6: state ← state XOR round Keys[round]
End for
// Final Key Mixing
Step 7: state ← state XOR round Keys[31]
Step 8: End
```

3.2 The RC6

A symmetric block cipher is considered to balance security and performance. A changeable number of iterations is implemented, with a maximum boundary of twenty, to ensure a resilient and protected cryptographic procedure. These iterations encompass intricate bitwise and arithmetic manipulations, such as additions and XOR operations. The flexibility in the number of iterations allows RC6 to familiarize itself with diverse security fundamentals, arresting a precise equilibrium between computational intricacy and cryptographic potency. This dynamic attribute distinguishes the original RC6. It offers flexibility in modifying its security constraints used for various applications, covering extremely secure settings to situations where a nuanced balance between efficiency and robust encryption is essential. The subsequent pseudo-code elucidates the process of the initial RC6 rounds [13]. The subsequent pseudo-code explains the procedure of the initial RC6 rounds.

Algorithm (3.2) expresses the pseudo-code of the original RC6 rounds:

```
Algorithm (3.2) RC6Round (A, B, C, D, S)

Input:
   A, B, C, D: 32-bit integers representing the current state
   S: Array of round keys
  w= 32 //Assuming a 32-bit block size
  mod= 2^w

Output:
  A, B, C, D: Updated state after applying the round function

Begin:
//Key Mixing
Step 1: B← (B + S[0]) MOD mod      // mod=2^Block_size
Step 2: D← (D + S[1]) MOD mod
// Main Loop (Repeating for r rounds)
Step 3: for i← 1 to r:                      //r: number of rounds
       t← (B * ((2 * B) MOD mod + 1)) MOD mod       //t: subsequent operations
       u← (D * ((2 * D) MOD mod + 1)) MOD mod     //u: subsequent operations
       A← (Circular Shift(XOR(A, t), w, u, 'L') + S[2 * i]) MOD mod    // L: circular shift operation
       C← (Circular Shift(XOR(C, u), w, t, 'L') + S[2 * i + 1]) MOD mod
         Swap (A, B, C, D)
Step 4: End for
    // Final Steps
Step 5: A← (A + S[Variables.t - 2]) MOD mod
Step 6: C← (C + S[Variables.t - 1]) MOD mod
Step 7: End
```

4. Comparison Factors and Evaluation Methods

In this paper, in order to evaluate encryption algorithms in cloud environment then make decision about suitable algorithm, the main factors are compared and their performance and security using NIST test. Based on various factors as shown in table 1 the encryption algorithms PRESENT and RC6 are compared [14]:

- *Key Size in Bits*: Assess the algorithms based on the key size options provided (e.g., 80, 128, 192, 256 bits) and their resistance to key-related attacks.

- *Block Size*: Evaluate the block size of each algorithm (e.g., 64 bits) and how it affects security and performance in cloud computing.

- *Cipher Classification*: Categorize each algorithm based on NIST's classification (e.g., block cipher, stream cipher) and ensure it matches your research objectives.

- *Computational Efficiency*: Evaluating the effectiveness of individual algorithms based on computational resources, is a crucial factor, particularly within cloud computing environments.

- *Encryption Time*: Measure the duration needed for encryption concerning individual algorithms, taking into account real-life scenarios and extensive data sets.

- *Decryption Time*: Measure the decryption time taken for individual algorithms, which is influential for cloud data access and processing.

- *Memory Usage*: Assess the memory expending of each algorithm to realize its resource requirements in the cloud environment.

- *Avalanche Effect*: Measures how variation in input data impacts the output to assess the algorithm's diffusion property. The Avalanche Effect can be quantified using the following equation (2):

$$\textbf{\textit{Avalanche Effect}} = \frac{\textbf{Number of differing bits}}{\textbf{Total number of bits in the output}} \times 100\% \qquad (2)$$

Where:

Number of differing bits: is the Hamming distance between the two output strings.

Total number of bits in the output: is the length of the output in bits.

The result is expressed as a percentage, showing the proportion of the output bits that changed.

- *Entropy*: Measures the randomness and unpredictability of the algorithms' output to ensure it is vital to security. The higher entropy values indicate a higher degree of randomness and unpredictability in the encrypted data, which is generally worthwhile for cryptographic security. The entropy H(X) of a random variable X (in this case, the output of a cryptographic algorithm) can be calculated using Shannon's entropy in equation (3):

$$H\,(X) = -\sum_{i=1}^{n} \mathbf{P}(\mathbf{xi})\ \mathbf{log2P}(\mathbf{xi}) \qquad (3)$$

Where:

P(xi): is the probability of each possible output xi.

n: is the number of possible outputs.

- *Energy Consumption*: Estimates the energy consumption of the algorithms, which is critical if the research includes energy-efficient cloud computing.

5. NIST Compliance Testing

NIST (National Institute of Standards and Technology) has developed several test suites and tools that are applicable to cryptographic algorithms to assess their performance and security. Each test suite focuses on different aspects of algorithm evaluation, such as randomness, entropy, key management, or block cipher modes [15].

5.1 The NIST Statistical Test Suite SP 800-22

The NIST Statistical Test Suite (NIST SP 800-22) is designed to assess the quality of random number generators, which are a critical element in cryptographic systems. The primary objective of the examinations is to assess the level of stochasticity exhibited by binary sequences that are produced by either hardware or software systems. Here are some advantages of each test [16]:

1. *Monbiot Test*: Checks the frequency of 0s and 1s in a binary sequence.

2. *Frequency within Block Test (Block Frequency Test):* Like the Monbiot test, this procedure involves dividing the given sequence into smaller blocks and subsequently testing each individual block for a balanced distribution of both 0s and 1s.

3. *Run Test*: Looks at the number and distribution of runs (a sequence of the same bit) in the binary sequence.

4. *Longest Run of Ones in a Block*: Measures the lengthiest occurrence of consecutive ones in sequence blocks and examines its conformity to a stochastic sequence.

5. *Binary Matrix Rank Test*: The rank of disjoint matrices from the sequence is evaluated.

6. *Discrete Fourier Transform (DFT)*: Transforms the sequence into the frequency domain to check for periodic patterns.

7. *Non-Overlapping Template Matching*: Checks for the occurrence of specific sequences or templates.

8. *Overlapping Template Matching Test*: Like to Non-Overlapping Template Matching but allows for overlapping occurrences.

9. *Maurer's "Universal Statistical" Test*: Determines the number of bits to evaluate between patterns that match.

10. *Linear Complexity Test*: Determines the linear feedback shift register's length (LFSR) needed to generate the sequence.

11. *Serial Test*: Examines the frequency of each and every plausible occurrence of overlapping 2-bit patterns.

12. *Approximate Entropy Test*: Measures the complexity of the sequence.

13. *Cumulative Sums (Cusum) Test*: Looks for deviations of the running sum of the sequence from what one might anticipate from a random succession.

14. *Random Excursion Test*: Examines the number of cycles in a random walk derived from the sequence.

15. *Random Excursion Variant Test*: Similar to the previous test but more generalized.

These tests can be applied to the output of an encryption algorithm to evaluate its randomness and, by extension, its suitability for providing secure, indistinguishable encryption. If you are doing a performance analysis based on NIST guidelines, you could apply a subset of these tests that are most relevant to your encryption algorithm and the specific requirements of your application.

6. Results

Depending on the comparison factor table 1 show, the results using data of size 112KB on the two algorithms.

Table 1- Comparative Analysis of The RC6 and PRESENT Encryption Algorithms with data size 112 KB.

Factors	RC6	PRESENT
Key Size	128	128
Block Size	64	64
Cipher Classification:	Block cipher	Block cipher
Encryption time	5.33394 seconds	6.93681 seconds
Decryption time	6.034850 seconds	6.19063 seconds
space time (Encryption)	1189.000000 bytes/second	41.000000 bytes/second
space time (Decryption)	901.000000 bytes/second	128.000000 bytes/second
Memory Used After Encryption	67.63 MB	41.27 MB
CPU Consumption- encryption	4.50 %	4.90 %
CPU Consumption decryption	4.10 %	4.20 %
Avalanche Effect	100%	99.00%
Entropy	99.98%	100.00%
Energy Consumption	0.901 joules	0.956 joules

Table 1 shows that RC6 excels in security due to its complex design and has been more extensively analyzed academically. PRESENT shines in efficiency and is better suited for resource-limited settings like IoT devices. RC6 offers more configuration flexibility and key size options, while PRESENT is simpler to implement correctly. While RC6 is generally more studied, PRESENT sees more adoption in lightweight applications. In summary, choose PRESENT for low-resource environments and RC6 for higher security and more configuration options.

Using throughput to evaluate the system's data processing efficiency about data size, It gives the number of bits/bytes that can be processed per second by the system [17], as shown in equation (4). Figure (1) achieves the throughput results for both algorithms with a data size of 112 KB.

Throughput (in KB per second) = (Data Size)/ (Encryption Time) (4)

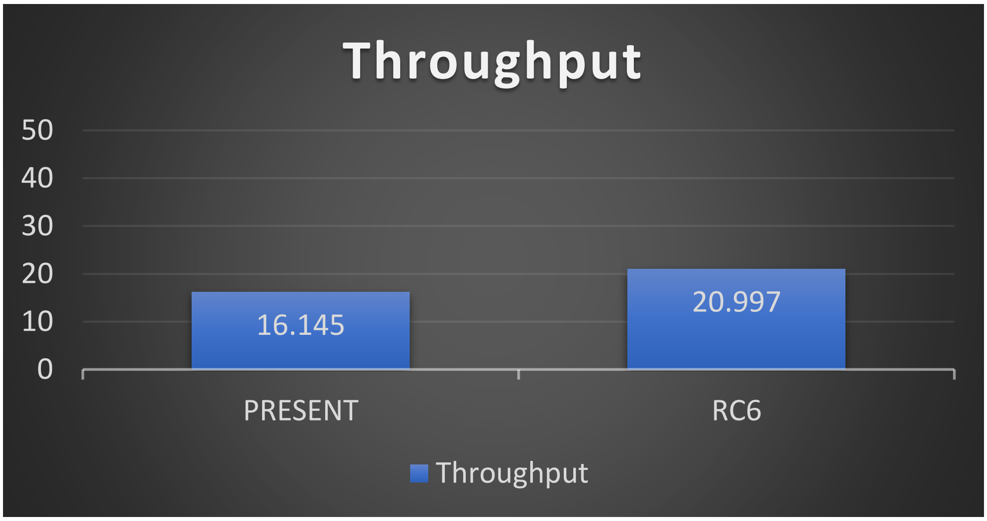

Fig. 1 - Throughput test for the RC6 and Present Algorithms.

For evaluation of the two-encryption algorithms, PRESENT and RC6, several key findings emerged from our performance analysis table 1 and table 2 show the results of comparison.

Table 2- NIST SP 800-22 Tests for Encryption algorithms.

Test	RC6	PRESENT
Monobit Test	0.48273366316155497	0.493072769772341
Frequency within Block Test	0.49140577999496426	0.45702808180647597
Run Test	0.5005939665120611	0.5040094778675822
Longest Run of Ones in a Block	0.5087660766434805	0.48662126639810976
Binary Matrix Rank Test	0.4787047563223393	0.24107119095114687
Discrete Fourier Transform (DFT)	0.4903327868568986	0.47766425691718983
NonOverlapping Template Matching	0.45251894224039524	0.44453743937143275
Overlapping Template Matching Test	0.03568226775422835	0.0
Maurer's "Universal Statistical" Test	0.99664912695655851	0.997788528202898
Linear Complexity Test	0.19403822604856973	0.4709518075396914
Serial Test	0.5050675179492101	0.54032764784546
Approximate Entropy Test	0.49884277041409025	0.4914512314747649
Cumulative Sums (Cusum) Test	0.5039465306987815	0.5100104669471928
Random Excursion Test	0.5472178294200204	0.5213654789010602
Random Excursion Variant Test	0.2512628976786059	0.4971234456001278

When judging the outcomes of examinations on encryption algorithms as per NIST SP 800-22, it is customary to deem a p-value of 0.01 or above as indicative of randomness with a confidence level of 99%. In cases where the p-value falls below 0.01, the sequence is deemed non-random, resulting in test failure.

From Table 2, to reach the result; for high-security needs, RC6 would generally be a better fit due to its higher security strength, especially if storing sensitive or critical data. For cost-efficiency, if the focus is more on saving computational resources, which translates to cost in a cloud environment, then PRESENT might be a more suitable choice. Regulatory Requirements: If the project needs to adhere to specific compliance standards, neither algorithm is NIST compliant.

Given that cloud environments generally have ample computational resources, and security is often a high priority, RC6 would likely be the better option for most cloud-based applications.

7. Discussion

In general, both RC6 and PRESENT have experienced extensive academic and professional scrutiny, which offers confidence to their security postures.

RC6 be likely to overtake PRESENT in terms of computational efficiency, confirmed by faster encryption and decryption times. Despite the fact that PRESENT achieves well in terms of avalanche effect also entropy, its active inefficiencies could attitude challenges in scalable or high-performance situations.

RC6 is more resource effectual, exploiting less memory and CPU for the duration of both encryption and decryption processes. In contrast, PRESENT's higher resource consumption may not make it the best select for cloud-based environments somewhere custom source can be a restriction.

Mutually algorithms use a 128-bit key size and a 64-bit block size, submission alike storage and complexity constraints in these respects.

- Avalanche Effect and Entropy:

Both algorithms shine in terms of avalanche effect and entropy, demonstrating robust data diffusion and randomness abilities. This is serious for guaranteeing the cryptographic strength of the algorithms.

The minor complexity time of PRESENT might make it easier to implement, but RC6’s reasonable complexity could be symbolic of a possibly more secure algorithm. RC6's speed and lower resource consumption also make it more appropriate for cloud environments that highlight performance.

These results emphasize the importance of considering explicit application necessities when selecting encryption algorithms. While NIST compliance confirms observance to established standards, balances between security and efficiency would be carefully evaluated.

The Figure (2) provides a summary of the performance features of the carefully chosen lightweight encryption algorithms, together with encryption speed, resource consumption, energy consumption, and NIST compliance. Let's match the encryption algorithms (PRESENT, and RC6) to Balancing Security and Performance:

- Security:

RC6 proposals a high level of security, making it perfect for safeguarding sensitive or dangerous data. Its security strength is flexible, thanks to a flexible key-size.

Considered as a lightweight cipher, PRESENT offers reasonable security, making it appropriate for a lesser amount of sensitive applications. It is less adaptable than RC6 in terms of key size and, thus, offers a narrower range of security strengths.

- Performance:

While RC6 is computationally exhaustive matched to lightweight ciphers, in cloud environments with plenty computational resources, this is generally not a concern.

Its key advantage dishonesties in its low computational necessities, making it very efficient. This is useful for real time applications or situations with resource limitations.

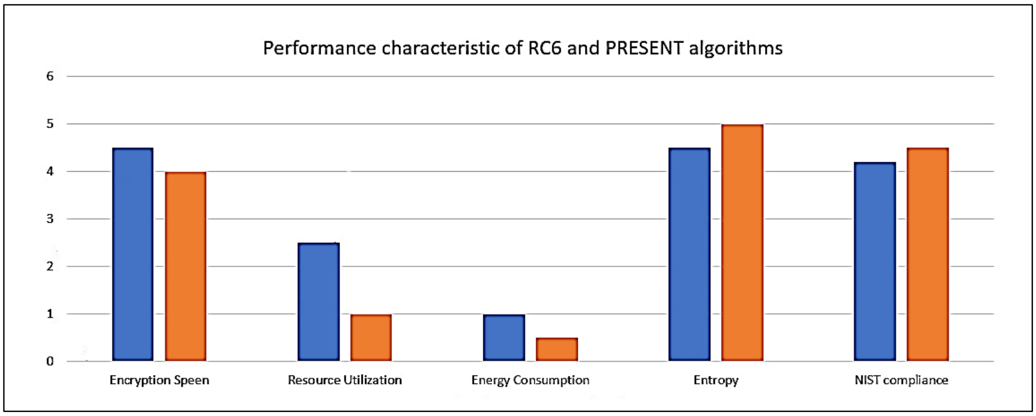

Fig. 2 - Performance characteristic of RC6 and PRESENT algorithms.

8. Conclusion

This paper on stimulating data security within cloud environments, carefully examined two encryption algorithms: RC6 and PRESENT. The conclusions from our in-depth performance analysis have lightened the special advantages and trade-offs associated with each algorithm.

NIST compliance was identified as an important factor, but it's worth noting that neither RC6 nor PRESENT are formally NIST-compliant. This should be considered especially in settings that are subject to strict regulatory oversight. When evaluating against common criteria such as security, performance, NIST compliance, scalability, and ease of implementation, both RC6 and PRESENT algorithms. For the given data size, a throughput of 20.997 surpasses 16.145. Throughput serves as a metric quantifying the volume of data that can be handled within a specific time frame. Elevated throughput figures signify enhanced performance, signifying a greater volume of data being handled within identical time constraints. Consequently, 20.997 demonstrates greater efficiency and efficacy in comparison to 16.145. In future Consider implementing hybrid encryption schemes that combine the strengths of multiple algorithms. Hybrid approaches combining symmetric and asymmetric encryption to establish a robust encryption scheme, allow you to achieve a balance between security and performance while maintaining compliance with NIST standards.

References

[1] Kumar, J. & Saxena, V. Cloud Data Security Through Bb84 Protocol And Genetic Algorithm. Baghdad Science Journal 19, 1445–1453 (2022).

[2] William Stallings. Cryptography And Network Security (Principles And Practice). (British Library Cataloguing-In-Publication Data, 2017).

[3] Rokan, J., Naser, N. M. & Naif, J. R. New Ultra-Lightweight Iot Encryption Algorithm Using Novel Chaotic System International Journal On 'Technical And Physical Problems Of Engineering' Ijtpe Journal New Ultra-Lightweight Iot Encryption Algorithm Using Novel Chaotic System. Issue 53, 253–259 (2022).

[4] Bassham, L. E. Et Al. A Statistical Test Suite For Random And Pseudorandom Number Generators For Cryptographic Applications. Https://Nvlpubs.Nist.Gov/Nistpubs/Legacy/Sp/Nistspecialpublication800-22r1a.Pdf (2010) Doi:10.6028/Nist.Sp.800-22r1a.

[5] Security and Privacy Controls for Information Systems and Organizations - NIST 800- 53 Rev.4. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r5.pdf (2013) doi:10.6028/NIST.SP.800-53r5

[6] Patil, P., Narayankar, P., Narayan, D. G. & Meena, S. M. A Comprehensive Evaluation Of Cryptographic Algorithms: Des, 3des, Aes, Rsa And Blowfish. In Procedia Computer Science Vol. 78 617–624 (Elsevier B.V., 2016).

[7] Barker, E., Roginsky, A. & Davis, R. Recommendation For Cryptographic Key Generation. Https://Nvlpubs.Nist.Gov/Nistpubs/Specialpublications/Nist.Sp.800-133r2.Pdf (2020) Doi:10.6028/Nist.Sp.800-133r.

[8] Podimatas, P. & Limniotis, K. Evaluating The Performance Of Lightweight Ciphers In Constrained Environments—The Case Of Saturnin. Signals 3, 86–94 (2022).

[9] Herman, M. Nist Cloud Computing Forensic Reference Architecture. Https://Nvlpubs.Nist.Gov/Nistpubs/Specialpublications/Nist.Sp.800-201.Ipd.Pdf (2023) Doi:10.6028/Nist.Sp.800-201.Ipd.

[10] D. S. Salman, J. R. Naif "Comparative Study Of Chaotic System For Encryption," Iraqi Journal for Computers and Informatics, Vol. 49, Issue 2, 2023, doi: 10.25195/ijci.v49i2.457.

[11] Mailewa, A. B. Et Al. Encryption Methods And Key Management Services For Secure Cloud Computing: A Review. Https://Www.Researchgate.Net/Publication/369777264.

[12] Bogdanov, A. Et Al. (2007). Present: An Ultra-Lightweight Block Cipher. In: Paillier, P., Verbauwhede, (Eds) Cryptographic Hardware And Embedded Systems - Ches 2007. Ches 2007. Lecture Notes In Computer Science, Vol 4727. Springer, Berlin, Heidelberg. Https://Doi.Org/10.1007/978-3-540-74735-2_31.

[13] R. Fan, T. Cui, S. Chen, C. Jin, And H. Zheng, "Multiset Structural Attack On Generalized Feistel Networks," Hindawi: Mathematical Problems In Engineering Journal, Vol. 2019, Doi: 10.1155/2019/2390462.

[14] William Stallings. Cryptography And Network Security (Principles And Practice). (British Library Cataloguing-In-Publication Data, 2017).

[15] About Nist. The National Institute Of Standards And Technology Https://Www.Nist.Gov/About-Nist (2022).

[16] Andrew Rukhin Et. Al. "Nist 800- 22 Rev. 1a: A Statistical Test Suite For Random And Pseudorandom Number Generators For Cryptographic Applications," Nist, Gaithersburg, Md 20899-8930, April 2010, Doi:10.6028/Nist.Sp.800-22r1a

[17] N. Goswami, B. Cao, and T. Li "Power-performance Co-optimization of Throughput Core Architecture using Resistive Memory" IEEE Conference: 19th International Symposium on High Performance Computer Architecture (HPCA), Feb. 2013, doi: I: 10.1109/HPCA.2013.6522331.