PSR: An Improvement of Lightweight Cryptography Algorithm for Data Security in Cloud Computing

Dr. P. Sri Ram Chandra[1]*, Dr.Syamala Rao P[2], Dr. Naresh K[3], Dr. Ravisankar Malladi[4]
Computer Science and Engineering, Shri Vishnu Engineering College for Women,
Bhimavaram, West Godavari District, Andhra Pradesh, India[1]
Information Technology, SRKR Engineering College, Bhimavaram, West Godavari District, Andhra Pradesh, India[2]
Department of Computer Science and Engineering,
TKR College of Engineering & Technology, Hyderabad, Telangana, India[3]
Department of CSE, Koneru Lakshmaiah Education Foundation,
Vaddeswaram, Guntur District, Andhra Pradesh-522302, India[4]

***Abstract*—Data security in cloud storage is a pressing concern as organizations increasingly rely on cloud computing services. Transitioning to cloud-based solutions underscores the need to safeguard sensitive information against data breaches and unauthorized access. Traditional cryptography algorithms are vulnerable to brute-force attacks and mathematical breakthroughs, necessitating large key sizes for security. Moreover, they lack resilience against emerging quantum computing threats, posing a significant risk to encryption. To tackle these issues, this study presents a novel lightweight cryptography algorithm named as PSR which is aimed at encryption so as to improve data security before storage in cloud systems. The proposed system converts 128 bit plaintext to cipher by employing techniques such as substitution, ASCII and hexadecimal conversions, block-wise transformations including Rail Fence, Grey Code, and XOR operations with random prime numbers. Notably, the proposed algorithm demonstrates superior performance with minimal runtime and memory usage, satisfying the avalanche effect criterion with a noteworthy efficacy in all executions and resistant to brute force attack.**

***Keywords*—Cryptography; cloud security; PSR; encryption; decryption; avalanche effect**

I. Introduction

Cloud computing plays a vital role in modern businesses by offering flexible and scalable solutions for storing and accessing data, facilitating innovation, and enhancing collaboration while reducing infrastructure costs and improving efficiency [1]. Ensuring data security in cloud computing is paramount, safeguarding sensitive information from unauthorized access and cyber threats. It fosters trust among users, promotes compliance with data privacy regulations, and mitigates the risks associated with data breaches. Robust security measures uphold the integrity and confidentiality of data, bolstering the reliability and credibility of cloud-based systems [2]. Traditional cryptography and lightweight cryptography represent two distinct approaches to securing data, each tailored to different needs and constraints. Traditional cryptography typically involves complex algorithms and protocols designed to provide high levels of security but may require significant computational resources and power consumption, making them less suitable for resource-constrained environments [3]. On the other hand, Lightweight cryptography is essential in cloud computing to optimize performance and resource usage, ensuring efficient data processing and secure communication across distributed networks while minimizing computational overhead [4]. In cloud computing, although current lightweight cryptographic algorithms provide efficiency, the ongoing development of new ones is crucial. Continuous development of new lightweight cryptographic algorithms in cloud computing ensures staying proactive against emerging threats and optimizing performance as environments evolve, supporting stronger, more resilient systems [5].

Encrypting data before storing it in the cloud enhances security by ensuring that only authorized parties with the decryption key can access the data, thus protecting against unauthorized access and data breaches. Additionally, encryption helps organizations meet regulatory compliance requirements and safeguards data integrity during transmission and storage [6]. In this research, the authors made progress in developing a novel cryptographic algorithm named as PSR, with the objective of encrypting data prior to its storage in cloud systems.

The subsequent sections of this paper are structured as follows: Section II outlines fundamental security requirements in cloud computing and explore research on lightweight cryptographic systems. Section III provides a brief overview of the proposed algorithm. Section IV presents the performance and security analysis of the proposed algorithm. Lastly, Section V offers conclusions and outlines future prospects.

II. Related Work

This section includes essential security needs in cloud computing along with research on lightweight cryptographic systems.

A. Security Requirements of Cloud Computing

Key security requirements in cloud computing, as outlined by NIST [7], include confidentiality, availability, integrity, authorization, authentication, accountability, and privacy.

Confidentiality involves restricting access to customer information to authorized individuals. Integrity ensures that information remains unaltered during processing or transmission, and that only authorized individuals can modify or delete it. Authentication verifies the identity of users accessing

*Corresponding Author.

data, typically through account security measures. Availability ensures that customer data and services are consistently accessible. Authorization controls access to data, allowing only authorized individuals to retrieve it [8-9].

B. Cloud Computing Security

Recent studies explore cloud computing security, focusing on cryptography. They analyze encryption algorithms like AES, IDEA, and DES, comparing symmetric and asymmetric methods. Parameters such as Block Size, Key Length, and Execution Time are evaluated for efficiency, especially in the cloud environment [10]. [11] conducted a study on major cloud service providers like Google (Google Drive) and Microsoft (Azure and OneDrive). They examined cryptographic algorithms commonly utilized in cloud computing, including modern cryptography, searchable encryption, homomorphic encryption, and attribute-based encryption (e.g., DES, 3DES, AES, RC6, and BLOWFISH). By combining multiple cryptographic techniques, they introduced a hybrid encryption approach to enhance cloud data security. Additionally, [12] compared IDAs, SHA-512, 3DES, and AES-256, focusing on on-premise data encryption and decryption.

C. Study of Lightweight Cryptography Systems

M. Usman *et al.,* [13] examined the Stable IoT (SIT) lightweight encryption algorithm, which utilizes a 64-bit block cipher and mandates data encryption with a 64-bit address. This approach incorporates elements of both the Feistel structure and a uniform substitution-permutation network. In study [14], a novel symmetric stream cipher, Ultramodern Encryption Standard (UES), is introduced for secure data transmission, utilizing prolic series numbers for key generation and binary/gray code operations for encryption and decryption. Sriram C P *et al.* introduced the Modular Encryption Algorithm (MEA) [15], a novel symmetric block cipher utilizing a tri-modular matrix for key generation and employing matrix operations, permutations, and substitutions for encryption and decryption processes. Authors in study [16] present a new symmetric stream cipher called the "Random Prime Key (RPK)" Algorithm, with an evaluation of its resilience against differential cryptanalysis and other pertinent factors, aiming for equilibrium between simplicity and security. In study [17], the "RECTANGLE" cryptosystem is outlined, designed for a 64-bit block size with key lengths of either 80 or 128 bits, and it executes 25 rounds. In study [18], the paper discusses a lightweight encryption algorithm for IoT devices, featuring a 64-bit block cipher and an 80-bit key for data encryption. In study [19], a lightweight cryptosystem features a 64-bit block size and 128-bit key, executed over 32 rounds with XOR operations and rotations. Its goal is hardware deployment in ubiquitous devices like wireless sensors and RFID tags, aiming for AES-level chip size but with faster performance.

III. The Proposed Algorithm

To enhance data security within cloud computing, the authors introduced a new lightweight cryptography algorithm named as PSR, aimed at encrypting data prior to storage in cloud systems. PSR encrypts 128-bit binary blocks using a 128-bit key through 10 encryption rounds, relying on mathematical functions for diffusion and confusion in each round. The system employs techniques like substitution, ASCII and hexadecimal conversions, block-wise transformations including Rail Fence and Grey Code, and XOR operations with random prime numbers. Additionally, it assesses the avalanche effect by comparing differing bits between the original plaintext and resulting cipher text.

A. Encryption Process

To safeguard sensitive data, the proposed algorithm PSR employs an encryption process that involves a sequence of cryptographic techniques. This process converts plain text into cipher-text while ensuring confidentiality and integrity. Here's a summary of the steps involved in one round of encryption process:

Input: Plain text message

```
Substitution Box Transformation:
   Define substitution_box mapping characters to substitutes
     substituted_text = ""
   for each character in input_text:
      substitute = substitution_box[character]
      append substitute to substituted_text
ASCII Conversion:
   ascii_values = []
   for each character in substituted_text:
      ascii_value = convert character to ASCII value
      append ascii_value to ascii_values
Hexadecimal Conversion:
   hexadecimal_values = []
   for each ascii_value in ascii_values:
      hexadecimal_value = convert ascii_value to hexadecimal
      append hexadecimal_value to hexadecimal_values
Hexadecimal to Binary Conversion:
   binary_values = []
   for each hexadecimal_value in hexadecimal_values:
      binary_value = convert hexadecimal_value to 8-bit binary
      append binary_value to binary_values
Block-wise Transformation:
   cipher_text = ""
   for each block in binary_values:
      rail_fence_1 = ""
      rail_fence_2 = ""
      for each bit in block:
         if position_of_bit is even:
            append bit to rail_fence_1
         else:
            append bit to rail_fence_2
      grey_code = generate_grey_code(block)
      left_shifted = left_shift(grey_code, 2)
      not_operation_result = apply_not_operation(left_shifted)
      new_substitution_box_result =
apply_new_substitution_box(not_operation_result)
      prime_number = generate_random_prime(min,max)
      prime_binary = convert prime_number to binary
      xor_result = perform_xor(new_substitution_box_result,
prime_binary)
      hexadecimal_result = convert xor_result to hexadecimal
      ascii_result = convert hexadecimal_result to ASCII
      character_result = convert ascii_result to character
      append character_result to cipher_text
```

Output: cipher_text

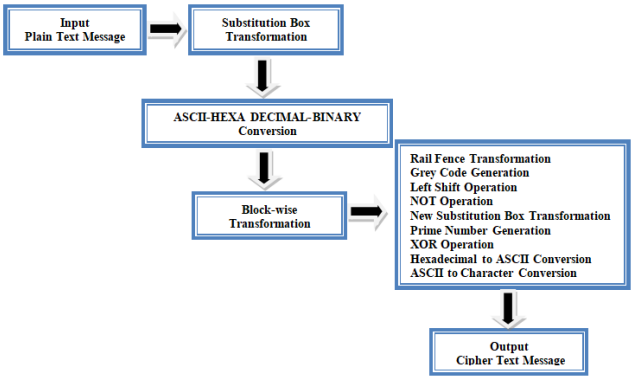

Fig. 1. PSR-Encryption process-flowchart.

The encryption process outlined in Fig. 1 begins by substituting characters in the plain text with predefined substitutes using a Substitution Box Transformation. ASCII Conversion converts substituted characters into ASCII values, followed by Hexadecimal Conversion for easier handling. Hexadecimal to Binary Conversion prepares data for block-wise transformation, where Rail Fence, Grey Code, Left Shift, and NOT operations are applied successively. Subsequent steps include a new Substitution Box Transformation, XOR operation with a key i.e., random prime number, and conversion back to cipher-text.

B. Key Exchange

The random prime keys used during the encryption process need to be exchanged with receiver so as to make use of them in decryption process. For safer exchange of keys between server and client, hybrid model that combines Diffie–Hellman (DH) and New-Hope (NH) is adopted [20].

C. Decryption Process

The decryption process that takes the cipher text as input and reverses the transformation steps to obtain the original plaintext message.

Input: Cipher text message

```
Character to ASCII Conversion:
  ascii_values = []
  for each character in cipher_text:
    ascii_value = convert character to ASCII value
    append ascii_value to ascii_values
Binary to Hexadecimal Conversion:
  hexadecimal_values = []
  for each 8-bit binary_value in ascii_values:
    hexadecimal_value = convert binary_value to hexadecimal
    append hexadecimal_value to hexadecimal_values
Block-wise Transformation:
  original_binary_values = []
  for each hexadecimal_value in hexadecimal_values:
    binary_value = convert hexadecimal_value to binary
    append binary_value to original_binary_values
Rail Fence and Grey Code Reconstruction:
  reconstructed_binary_values = []
  for each block in original_binary_values:
    not_operation_result = apply_not_operation(block)
    new_substitution_box_result =
reverse_apply_new_substitution_box(not_operation_result)
    xor_result = perform_xor(new_substitution_box_result,
prime_binary)
    reconstructed_binary_values.append(xor_result)
Binary to ASCII Conversion:
  decrypted_ascii_values = []
  for each binary_value in reconstructed_binary_values:
    decrypted_ascii_value = convert binary_value to ASCII
    append decrypted_ascii_value to decrypted_ascii_values
ASCII to Character Conversion:
  original_text = ""
  for each ascii_value in decrypted_ascii_values:
    character_result = convert ascii_value to character
    append character_result to original_text
```

Output: original_text

IV. Results and Discussion

To validate the proposed algorithm, comparative analysis and a series of security experiments were conducted to gauge the effectiveness of the coding scheme, focusing on evaluation metrics including processing time, confusion and diffusion, avalanche effect.

A. Comparative Analysis of Proposed Encryption Algorithm

A comparison between the proposed method and current encryption techniques mentioned in Table I which reveals notable differences. While existing methods like DES, AES, Blowfish, and LED utilize various structures and operations, the proposed technique introduces a distinct approach. Unlike DES's limited block and key sizes or AES's variable configurations, the proposed technique offers a fixed 128-bit block and key size. Notably, PSR introduces unique mathematical operations like Rail-fence and Grey code conversions, enhancing its security. With a focus on security, PSR demonstrates a highly secure approach, surpassing the proven inadequacies of DES while matching or exceeding the security levels of AES, Blowfish, and LED as shown in Table I.

B. Processing Time

In cryptography, "processing time" is crucial, determining how long it takes to perform cryptographic tasks, impacting the speed of secure communication. Less processing time in cryptography algorithms is important for ensuring efficient and timely secure communication. The data presented in Table II represents the average encryption time obtained from five consecutive experimental runs. Fig. 2 illustrates that the proposed PSR algorithm consistently outperforms or matches the processing times of established algorithms like DES, AES, Blowfish, and LED across different file sizes. This underscores the superior efficiency and competitive advantage of the PSR algorithm in cryptographic operations.

C. Security Analysis

The proposed algorithm enhances the security of data before it gets stored onto the cloud by bolstering confidentiality and integrity while ensuring accessibility when needed. In terms of security, the PSR cryptographic algorithm can withstand well-known threats like weak key attacks, symmetric properties, related-key attacks, and differential and linear cryptanalysis [25, 26].

TABLE I. COMPARATIVE ANALYSIS OF PROPOSED ENCRYPTION ALGORITHM WITH EXISTING TECHNIQUES

Algorithm	**DES[21]**	**AES[22]**	**Blowfish[23]**	**LED[24]**	**Proposed Algorithm-PSR**
Structure	Feistel	Substitution-Permutation	Feistel	Feistel + SP	Feistel + SP
Block Size (bits)	64	128	64	64 or 128	128
Key Size (bits)	56	128, 192, 256	32–448	64 or 128	128
No. of Rounds	16	10, 12, 14	16	Variable	10
Key Space	256	2128, 2192, or 2256	232 – 2448	264 , 2128	2128
Mathematical Operations	Permutation, XOR, Shifting, Substitution	XOR, Mixing, Substitution, Shifting, Multiplication, Addition	XOR, Mixing, Substitution, Shifting	XOR, rotations, 2n mod addition, substitution	Substitution, Rail-fence, Binary and Grey code conversions, Shifting, NOT, XOR with Prime Number
S-P Structure	8 S-Box	1 S-Box	4 S-Boxes	4 S-Boxes	2 S-Boxes
Security Rate	Proven inadequate	Secure	Secure	Secure	Highly Secure

TABLE II. COMPARATIVE ANALYSIS OF PROPOSED ALGORITHM'S PROCESSING TIME AND EXISTING CRYPTOGRAPHY ALGORITHMS

Algorithm	**DES[21]**	**AES[22]**	**Blowfish[23]**	**LED[24]**	**Proposed Algorithm-PSR**
File Size (KB)	**Processing time (seconds)**				
28	0.0011	0.0014	0.0012	0.019	0.013
29	0.017	0.016	0.015	0.029	0.0152
210	0.028	0.029	0.031	0.0548	0.0352
213	0.29	0.29	0.24	0.62	0.45
214	0.945	0.805	1.06	1.45	0.789
215	1.91	1.74	2.01	2.53	1.65

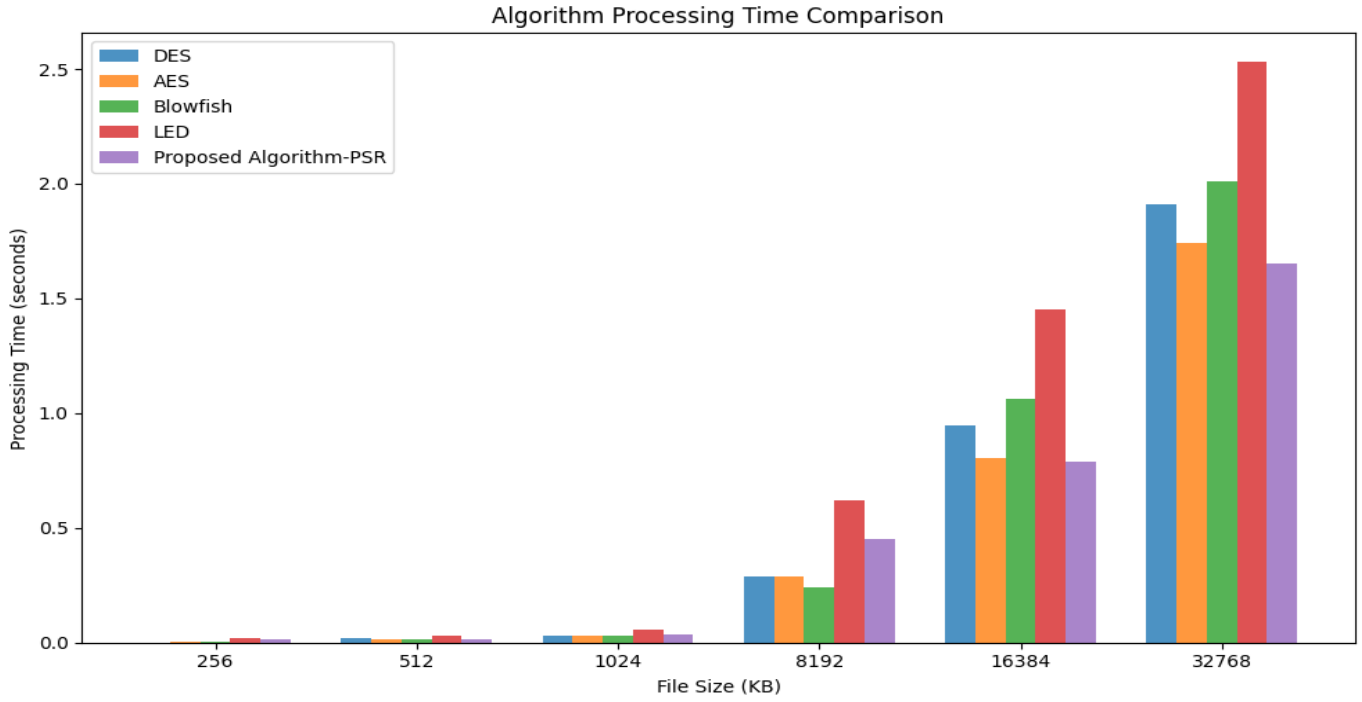

Fig. 2. Processing times of proposed algorithmm PSR cryptography algorithm and existing techniques.

$$Percentage\ of\ Avalanche\ Effect = \left(\frac{Number\ of\ bits\ flipped\ in\ the\ cipher\ text}{Number\ of\ bits\ present\ in\ the\ cipher\ text}\right) \times 100$$

D. Impact of Avalanche Effect-SPAC and SKAC

The algorithm must adhere to the Strict Plaintext Avalanche Criterion (SPAC), which implies that even a minor alteration in

the plaintext, while keeping the key constant, should lead to substantial changes in the resulting cipher text. Similarly, it should also meet the Strict Key Avalanche Criterion (SKAC), meaning that with the plaintext fixed, any slight modification in the key should produce significant variations in the generated cipher text [27]. The effectiveness of the PSR cryptography algorithm's security was assessed using SPAC and SKAC. Table III presents the outcomes of this assessment for a fixed plaintext ("Cryptography") across varying keys.

TABLE III. (A) SKAC ANALYSIS OF PSR CRYPTOGRAPHY ALGORITHM

Fixed Plaintext			
Test case	**Cipher-text (128 bits)**	**Number of Bits Changed**	**Avalanche effect (%)**
1	ùÂ¿ØbQDþ"Š /	69	53.9
2	ºs4ÂROpìÆøe	74	57.59
3	¡"WhÚ´Fp*¹	71	55.41
4	7hÄB¿ænðD¿	78	60.62
5	bEÒÜ»Db®y	80	62.64
6	å2Ù>¸Â¯D2¨êe	86	67.56
7	Ed¿Ø€ ¤Õ,à>P³	109	85.83
8	³^ØÊµØÆhê	64	50.09
9	OðE\öC^ì •	92	72.54
10	7º©&:Â[ì¨@®³	99	77.91
Average SKAC			64.40

(B): SPAC ANALYSIS OF PSR CRYPTOGRAPHY ALGORITHM

Fixed key			
Plaintext	**Cipher-text (128 bits)**	**Number of Bits Changed**	**Avalanche effect (%)**
Algorithm	Då_®þ¼Ö®s	70	51.66
Percentage	&®Â½|Áê	71	52.58
Avalanche	n*êEØ¤(|	77	57.22
Cloud Security	=oX÷Ë#Âèo©&,	92	69.10
Computing	QòkãbT¥	86	64.44
Brute Force Attack	1Â[J,eZÖª=&]pÂh´=²	83	62.08
Diffusion	gh|ÄtÖjþ	79	58.98
Confidentiality	åê8Â • ø&¶¸b7Äv¿	92	68.99
Integrity	¦V@è#ð|>5	95	71.66
Light weight	@9*#pÖÇ8v	86	64.56
Average SPAC			62.12

The results from the Tables III (A) and III (B) reveal that the percentages for SKAC (Strict Key Avalanche Criterion) and SPAC (Strict Plaintext Avalanche Criterion) are 64.40% and 62.12%, respectively. These thresholds demonstrate the algorithm's effectiveness in achieving a notable avalanche effect, which is critical for ensuring strong diffusion and enhanced security. Among the various test cases evaluated, the PSR algorithm stands out for delivering the highest avalanche percentages, showcasing its superior performance. This makes it particularly well-suited for encrypting the texts before storing them onto cloud systems, where data confidentiality, integrity, and resilience against brute force attacks are well maintained. With its ability to ensure high levels of diffusion, the PSR algorithm is an excellent choice for protecting sensitive data with respect to cloud-based environments.

E. Confusion and Diffusion

Confusion and diffusion, concepts initially explored by Shannon [28], are fundamental to encryption, aiming to complicate the relationship between encrypted text and keys. Proposed encryption technique employs operations such as substitution, ASCII and hexadecimal conversions, block-wise transformations including Rail Fence and Grey Code, and XOR operations with random prime numbers. Altering a single letter in the original text impacts numerous sections of the encrypted text, while each encryption of identical text generates a varied outcome, enhancing both complexity and security. Consequently, our approach seamlessly integrates the fundamental concepts of confusion and diffusion.

F. Resistant to Brute Force Attack

A brute force attack exhaustively tests all potential combinations to compromise encryption keys. The PSR cryptography algorithm employs a 128-bit binary key, leading to 2^{128} possible key combinations, guaranteeing unique keys for each encryption process.

V. CONCLUSION AND FUTURE SCOPE

Ensuring data security prior to its storage in the cloud has become increasingly crucial. Despite the existence of numerous cryptography algorithms aimed at bolstering data protection, there remains a significant demand for innovative approaches. This paper presents a novel cryptography algorithm, denoted as PSR, which operates on 128-bit plaintext using a 128-bit key to produce 128-bit cipher text. It draws inspiration from the architectural models of Fiestal and SP. The encryption method under consideration utilizes various operations, including substitution, ASCII and hexadecimal conversions, block-level transformations such as Rail Fence and Grey Code, as well as XOR operations involving random prime numbers. Despite boundaries like fixed key length and computational overhead, the PSR algorithm ensures high security with innovative techniques such as prime-based XOR, Rail Fence, and Grey Code. The experimental findings presented in Table I clearly indicate that the PSR algorithm, as proposed, offers a high level of security. Furthermore, Table II illustrates shorter processing times compared to current algorithms, affirming its applicability even in resource-limited environments. Table III data shows PSR cryptography excels in SKAC and SPAC, averaging 64.40% and 62.12%. The PSR algorithm's responsiveness to input variations increases output randomness, a desirable trait in cryptographic algorithms, thwarting attackers' predictions. Future work can focus on benchmarking PSR in real-world scenarios to evaluate its scalability and seamless integration with diverse cloud systems.

REFERENCES

[1] Pazun, Brankica. (2018). Cloud Computing influence on modern business. Serbian Journal of Engineering Management. 3. 60-66. 10.5937/SJEM1802060P.

[2] Soofi, Aized & Khan, M & Amin, Fazal-e. (2014). A Review on Data Security in Cloud Computing. International Journal of Computer Applications. 94. 975-8887. 10.5120/16338-5625.

[3] Tankard, C. Encryption as the cornerstone of big data security. Netw. Secur. 2017, 2017, 5–7.

[4] K. Huang, X. Liu, S. Fu, D. Guo, M. Xu, A lightweight privacy-preserving CNN feature extraction framework for mobile sensing, IEEE Trans. Dependable Secure Comput. 18 (3) (2019) 1441–1455.

[5] Thabit, Fursan & Alhomdy, Sharaf & Al-ahdal, Abdulrazzaq & Jagtap, Prof. (2021). A New Lightweight Cryptographic Algorithm for Enhancing Data Security In Cloud Computing. Global Transitions. 2. 10.1016/j.gltp.2021.01.013.

[6] Belguith, Sana. (2015). Enhancing Data Security in Cloud Computing Using a Lightweight Cryptographic Algorithm", The Eleventh International Conference on Autonomic and Autonomous Systems.

[7] P. Mell, T. Grance, The NIST definition of cloud computing - SP 800-145, NIST Spec. Publ. (2011), doi: 10.1136/emj.2010.096966 .

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

[9] Stallings, W. (2017). "Cryptography and Network Security: Principles and Practice." Pearson.

[10] D.S. Abd Elminaam , H.M.A. Kader , M.M. Hadhoud , Evaluating the performance of symmetric encryption algorithms, Int. J. Netw. Secur. (2010).

[11] J.R.N. Sighom, P. Zhang, L. You, Security enhancement for datamigration in the cloud, Futur. Internet (2017), doi: 10.3390/fi9030023.

[12] D.P. Timothy, A.K. Santra, A hybrid cryptography algorithm for cloud computing se- curity, 2017 International conference on Microelectronic Devices, Circuits and Sys- tems (ICMDCS), Vellore (2017) 1–5, doi: 10.1109/ICMDCS.2017.8211728 .

[13] M. Usman, I. Ahmed, M. Imran, S. Khan, U. Ali, SIT: a lightweight encryption algorithm for secure internet of things, Int. J. Adv. Comput. Sci. Appl. (2017), doi: 10.14569/ijacsa.2017.080151.

[14] P. Sri Ram Chandra, G.Venkateswara Rao,G.V.Swamy,_Ultramodern Encryption Standard Cryptosystem using Prolic Series for Secure Data Transmission', International Journal of Latest Engineering Research and Applications (IJLERA) ISSN: 2455-7137 Volume – 02, Issue – 11, November – 2017, PP – 29-35.

[15] P.Sri Ram Chandra, Dr. G.Venkateswara Rao and Dr.G.V.Swamy, Modular Encryption Algorithm for Secure Data Transmission Int. J. Sc. Res. In Network Security and Communication ISSN: 2321-3256 Volume-6, Issue-1, February 2018.

[16] U.Bhanu Prasanna and Dr.P.Sri Ram Chandra, Data Security using Efficient Cryptosystem, TEST Engineering and Management, ISSN: 0193-4120, 15355-15360, January-February2020, The Mattingley Publishing Co., Inc.

[17] W. Zhang, Z. Bao, D. Lin, V. Rijmen, B. Yang, I. Verbauwhede, RECTANGLE: a bit- slice lightweight block cipher suitable for multiple platforms, Sci. China Inf. Sci. (2015), doi: 10.1007/s11432-015-5459-7.

[18] A.H.A. Al-ahdal , G.A. Al-rummana , G.N. Shinde , N.K. Deshmukh, A Robust Lightweight Algorithm for Securing Data in Internet of Things Networks, sustainable Communication Networks and Application. Lecture Notes on Data Engineering and Communications Technologies, vol 55. Springer, (2021).

[19] Z. Gong, S. Nikova, and Y.W. Law, "KLEIN: a new family of lightweight block ci- phers, "2012, doi: 10.1007/978-3-642-25286-0_1.

[20] Hussein, A.I. (2023). Hybrid: (NH–DH) a New Hope and Diffie–Hellman for Transmission Data in Cloud Environment. In: Swaroop, A., Kansal, V., Fortino, G., Hassanien, A.E. (eds) Proceedings of Fourth Doctoral Symposium on Computational Intelligence . DoSCI 2023. Lecture Notes in Networks and Systems, vol 726. Springer, Singapore. https://doi.org/10.1007/978-981-99-3716-5_66

[21] M.A. Wright, The advanced encryption standard, Netw. Secur. (2001), https://doi.org/10.1016/S1353-4858(01)01018-2.

[22] A.U. Rahman, S.U. Miah, S. Azad, Advanced encryption standard, in: Practical Cryptography: Algorithms and Implementations Using Cþþ, 2014.

[23] M.N. Valmik, P.V.K. Kshirsagar, "Blowfish Algorithm," IOSR J. Comput. Eng. (2014), https://doi.org/10.9790/0661-162108083.

[24] G. Bansod, N. Raval, N. Pisharoty, Implementation of a new lightweight encryption design for embedded security, IEEE Trans. Inf. Forensics Secur. (2015), https://doi.org/10.1109/TIFS.2014.2365734.

[25] M. Usman, I. Ahmed, M. Imran, S. Khan, U. Ali, SIT: a lightweight encryption algorithm for secure internet of things, Int. J. Adv. Comput. Sci. Appl. (2017), doi: 10.14569/ijacsa.2017.080151.

[26] A.H.A. Al-ahdal , G.A. Al-rummana , G.N. Shinde , N.K. Deshmukh , A Robust Lightweight Algorithm for Securing Data in Internet of Things Networks, ustain- able Communication Networks and Application. Lecture Notes on Data Engineering and Communications Technologies, vol 55. Springer, (2021).

[27] Norman D. Jorstad.: Cryptographic Algorithm Metrics, January 1997.

[28] Shannon, C. E. (1949). Communication theory of secrecy systems. Bell System Technical Journal, 28(4), 656-715.

AUTHORS' PROFILE

Dr. P. Sri Ram Chandra serves Shri Vishnu Engineering College for Women, bringing a wealth of academic and professional experience. He holds a B.Tech from Andhra University and both an M.Tech and a Ph.D. from GITAM University. Dr. PSR has made significant contributions to his field, including the publication and review of books. He has also delivered guest lectures under the AICTE-STTP program and evaluated doctoral theses. Dr. PSR has an extensive record in academic service, having reviewed numerous research articles. His teaching career spans over a decade, during which he achieved numerous technical and research certifications. In addition to his academic achievements, Dr. PSR has published many patents and research publications. His research interests include, cryptography and information security, Theory of Computations. Detailed Profile: https://sites.google.com/view/dr-psr

Dr. Syamala Rao P. secured his Ph.D degree from Acharya Nagarjuna University. He is a topper in academics and secured gold medal in his M.Tech. He has 20+ years of experience in Academics and he is currently working as an Associate Professor in Information Technology department of S.R.K.R Engineering College, Bhimavaram. His research areas are cryptography, Machine Learning, Artificial Intelligence and Mining.

Dr. NARESH K is an Assistant Professor in the Department of CSE at TKR College of Engineering and Technology, Autonomous, Hyderabad. He received his Ph.D. in Computer Science & Engineering from the NIILM UNIVERSITY, Haryana in2023. M.Tech in Software Engineering, Jagruthi institute of Engineering and Technology, JNTU HYDERABAD in 2012. His research interests in cryptography, Machine Learning, Deep Learning and Computer Networks.

Dr. M. RaviSankar, working as an Associate Professor in Department of Computer Science and Engineering, K.L. deemed to be University, Guntur dist., Andhra Pradesh, India. He has 24 years of teaching experience. Dr. Ravi has received Excellence in Research Award and Best Senior Faculty Award. He has Published 5 Patents and he has published more than 20 articles in Scopus, SCI, WOS and International Journal., His areas of specializations are cryptography, Data Mining and Artificial Intelligence.