Int. Jrl. of **Human Computations & Intelligence**
Vol. 02, Issue 05, 2023
ISSN: 2583-5696

RESEARCH ARTICLE

Comparative Analysis of Cryptographic Algorithms

Samuel D Jonathan[1] . Jonathan S Paul[1] . Naveen Chandra Gowda[1]. Ambika B J[1]. Kiran Kumar P N[2]

[1]School of Computer Science and Engineering, REVA University, Bengaluru, India.

[2]Department of Computer Science and Engineering, S J C Institute of Technology, Chikballapur, India.

Received: 27 April 2023 / Revised: 16 May 2023 / Accepted: 08 June 2023

DOI: 10.5281/zenodo.8068102

Abstract – Secure communication and information exchange in the presence of adversaries is a critical issue in today's digital age. Cryptography is becoming increasingly important in our modern world, as we rely more and more on secure communication and data transfer. In this paper, we explore the performance of several widely-used cryptographic algorithms. Our aim is to provide a useful resource for anyone seeking to improve their understanding of encryption and the factors that affect its effectiveness. Through our experiments, we hope to shed light on the strengths and weaknesses of different algorithms and to help researchers and practitioners make informed decisions about which techniques to use in their work. This research paper aims to provide a comparative analysis of the cryptographic algorithms Serpent, Two fish and Salsa20. This paper also analyzes parameters like key size, block size, number of rounds, and throughput of the algorithms to determine the efficiency of the cryptosystems.

Index Terms – Cryptography, AES, Twofish, Serpent, Salsa20, Security, Encryption, Decryption, Security, Simulation, Comparison, Block Ciphers.

I. INTRODUCTION

In today's digital age, securing information and communication has become a crucial aspect of modern society. With the increasing reliance on technology and the internet, sensitive data is vulnerable to interception, theft, and manipulation by malicious actors. This is where cryptography, the science and art of securing communication, comes into play. Cryptography is a technique used to protect messages and data by transforming them in a way that makes them unintelligible to unauthorized parties [1]. By converting plain text into ciphertext, cryptography ensures the confidentiality, integrity, authenticity, and non-repudiation of data. The use of cryptography has become an integral part of many applications and systems, including secure messaging, online banking, e-commerce, and data storage. Further cryptographic based encryption mechanisms are used for securing the image data [2][3]. In this paper, we will perform a comparative analysis of various cryptographic algorithms, exploring their

strengths and weaknesses, and providing insights on how to select the appropriate algorithm for specific use cases. Encryption schemes are divided into two groups, block ciphers and stream ciphers. Block ciphers divide the input message into blocks of fixed size, and perform encryption on the blocks separately. The encrypted blocks are combined to generate the overall cipher text. Stream ciphers, however, treat the input message as a stream of bits and encrypt the entire message at once, unlike block ciphers.

SERPENT

The Serpent cipher uses a single key for encryption and decryption in the algorithm that was designed in 1998 by a team of cryptographers including Ross Anderson, Eli Biham, and Lars Knudsen [4]. It is a block cipher that encrypts fixed-sized blocks of plaintext using a secret key. The algorithm uses a substitution-permutation network (SPN) structure and is designed to provide high security with a low risk of potential weaknesses. Serpent cipher has become popular in numerous applications such as disk encryption, VPNs, and secure communications, due to its high-security level, efficient implementation, and versatility. As a result, it is considered to be a reliable encryption algorithm for a wide range of use cases.

The Serpent cipher has a block size of 128 bits and supports key sizes of 128, 192, and 256 bits [5][6]. It is based on the use of S-boxes, which are tables that replace input values with output values based on a certain algorithm. Each round uses 32 copies of the same 4-bit to 4-bit S-box [7]. The S-boxes used in the Serpent cipher are carefully designed to provide a high degree of nonlinearity and resistance against differential and linear cryptanalysis attacks.

TWOFISH

The block cypher algorithm Twofish uses a variable-length key that can be up to 256 bits long. The cypher consists of a 16-round network with a fixed 4-by-4 maximum distance separable matrix, four 8-by-8-bit S-boxes that depend on the keys, and a bijective F function. The eight sub-keys K0-K7 are used to XOR the input and output data. Input and output whitening are the names given to these XOR operations [8]. The eight sub-keys K0 through K7 are used to XOR the input and output data in the two-fish technique. Both the input and output of these X-AND operations are whitened. Key-dependent S-boxes, MDS matrices, and pseudo-Hadamard transforms (PHT) are a few of the five various kinds of component operations that make up the F-function [9]. The architecture of the round function and the key scheduling allows a number of trade-offs between speed, software size, key setup time, gate count, and memory.

SALSA20

Salsa20 takes a 256-bit key and a 64-bit nonce and uses them to generate a 270-byte stream. To encrypt a plaintext of b bytes, Salsa20 XORs the first b bytes of the stream with the plaintext, and discards the remaining bytes. To decrypt a ciphertext of b bytes, Salsa20 XORs the first b bytes of the stream with the ciphertext [10].

The stream is created in 64-byte (512-bit) blocks, each of which is a unique hash of the key, nonce, and block number. Since there is no chaining between blocks, the stream can be arbitrarily accessed and any number of blocks can be computed concurrently. Unlike other encryption algorithms, Salsa20 does not have any hidden preprocessing costs. Each block uses the key and nonce directly as input, without any additional preprocessing. This makes Salsa20 a very efficient and effective encryption algorithm for a wide range of applications. Cryptographic algorithms are very important to achieve privacy and data confidentiality with access control between the communicating parties [11] and in turn maintain the trust among the devices [12].

II. LITERATURE SURVEY

Here we provide the brief literature of considered cryptographic algorithms and their usages.

Redesigning the Serpent Algorithm By Pa-Loop And Its Image Encryption Application

The limitations of existing image encryption techniques that use their own proposed structure, often sacrifice speed and security. Despite its initial success, the DES algorithm lost popularity due to its shorter key length and its original design for hardware enciphering, rather than software encryption [13]. This led to the development of the AES. A new method for image encryption using the Serpent cipher algorithm in a Feistel network structure is used. While previous research has utilized various image encryption techniques, including BCH Codes, Elliptic curve, Cyclic codes, QFT, SPN network, Polynomial mapped, and Mobius transformation, the paper took a different approach by incorporating power associative loop structure into the modified Serpent algorithm. The proposed method is demonstrated to effectively encrypt images, offering a promising solution to image protection.

This modified version of the Serpent encryption scheme, where the S-box construction is different and is developed using Power-Associative (PA) loops. The proposed scheme uses a 128-bit key and PA loops of order 256, providing a larger key space than the extended binary Galois field. This makes it harder for an attacker to break the system, even if they have knowledge of the key but not the loop. The noncommutative nature of the proposed mathematical system also enhances its security. The scheme is applicable to both text and image encryption and has been tested through various analyses, showing its effectiveness in real-world scenarios [14].

The Saturation Attack - A Bait For Twofish

In the paper, the concept of a "saturation attack" is introduced, which takes advantage of a permutation p over w-bit words. The attack, at the time of the paper's research, was considered the best attack against the Twofish cipher. The set of outputs is the same as the set of inputs when p is applied to all 2w disjoint words. The use of saturation attacks on reduced-round Twofish block cyphers, with up to seven rounds with full whitening or eight rounds without whitening, is then explored in the study. These attacks can be up to 2-4 times faster than exhaustive search and require up to 2127 chosen

plaintexts. The attacks use key-independent distinguishers for up to six rounds of Twofish and rely heavily on the saturation properties.

The Twofish cipher is made up of 16 rounds and features two-sided whitening. Attackers are only able to break one half of the cypher when using saturation attacks on reduced-round varieties of Twofish with up to seven rounds with complete whitening or eight rounds without whitening at the conclusion (i.e., half of the cypher). This means that Twofish still maintains a reasonable security margin against the attack [15]. It directly can affect communication in various applications like VANET, email, smart applications [16] [17].

A Lightweight Cipher Based On Salsa20 For Resource-Constrained Iot Devices

The paper incorporates a Salsa20-based cipher to provide security to IoT applications. Despite having a considerable amount of benefits, IoT has characteristics that make it vulnerable to security threats. In addition, the limitations of computing and energy resources in IoT devices constrain their ability to implement current ciphers. The paper covers the use of the Generador de Bits Pseudo Aleatorios (GBPA) cipher that is based on Salsa20 that was designed to meet the low computing requirements of IoT devices. Its implementation allows for IoT devices to attain a higher level of security, providing greater privacy to user's and protection against damaging attacks to the systems. From the research it can be seen that by making a few improvement to the Salsa20 cipher, memory usage and computing requirements (namely, CPU cycles and power consumption) can be reduced to better match IoT devices. The paper also mentions the security of the Salsa20 cipher itself and how it's security can only be compromised by reduction in the number of rounds [18] [19].

III. COMPARATIVE ANALYSIS OF ALGORITHMS

After studying the various techniques used to perform encipherment, we have done the comparison based on the following important factors: Input data size: This parameter refers to the size of the data that is being encrypted or decrypted by the algorithm. It is an important parameter to consider because different algorithms may have different performance characteristics depending on the size of the input data. For example, an algorithm that performs well with small input sizes may not necessarily perform well with larger input sizes. In the comparative analysis, we varied the input data size to evaluate how each algorithm performs with different input sizes.

Time: This parameter refers to the time taken by the algorithm to encrypt or decrypt the input data. It is an important parameter to consider because the speed of the algorithm directly affects the usability of the encryption technique. A slow algorithm can be impractical to use in real-world scenarios. In the comparative analysis, we measured the time taken by each algorithm to encrypt and decrypt the input data. Throughput: This parameter refers to the amount of data that can be encrypted or decrypted by the algorithm in a given amount of time. It is an important parameter to consider because it indicates the efficiency of the algorithm. A high throughput algorithm can handle large amounts of data quickly and efficiently. In the comparative analysis, we measured the throughput of each algorithm by calculating the amount of data that could be encrypted or decrypted per unit time.

The comparison parameters are shown in Table 1.

Table 1: Comparison of features

Features	SERPENT	TWOFISH	SALSA20
Key Used	same key for encryption and decryption	same key for encryption and decryption	same key for encryption and decryption
Encryption scheme	Block cipher	Block cipher	Stream cipher
Block size	128 bits	128 bits	each bit is treated individually
Key Size	128/ 196/ 256 bits	variable (up to 256 bits)	256 bits
Rounds	32	16	20

Simulation Analysis

This section of the paper presents the results obtained after simulating the cryptographic algorithms and testing them with various data input sizes measured in kilobytes. The study takes into account the encryption and decryption times for each algorithm. The purpose of the experiment is to compare the algorithms based on their performance in terms of time and throughput. To conduct the experiment, we generated random input data of various sizes ranging from 48KB to 5344KB. The time taken to encrypt and decrypt the data was measured for each algorithm, and the throughput was calculated based on the amount of data that could be processed in a given time. The experiment provides insights into the strengths and weaknesses of each algorithm and helps in making an informed decision about selecting an appropriate algorithm for a particular application. The average of 5 iterations were taken for each input sample. NOTE: Serpent and Twofish algorithms were written in Python2 whereas Salsa20 was written in Python3. The encryption time is shown in Table 2 and decryption time in Table 3.

Table 2: Encryption time(milliseconds) of different data packet size(KB)

Input size (KB)	Serpent	Twofish	Salsa20
48	243.31	77.63	0.16
64	340.08	101.09	0.22
96	532.25	162.07	0.32
240	1311.14	416.61	0.83
320	1681.79	608.34	1.19
560	2966.38	1252.55	2.75
896	4769.74	3047.90	4.09
5344	29320.63	82054.55	21.16
Throughput (KB/Sec)	187.256	468.711	265529.066

Table 3: Decryption time(milliseconds) of different data packet size(KB)

Input size (KB)	Serpent	Twofish	Salsa20
48	243.25	76.89	0.14
64	331.28	102.73	0.19
96	528.69	166.01	0.27
240	1289.41	430.98	0.69
320	1858.94	597.76	1.00
560	3052.43	1257.60	1.71
896	5099.78	3046.67	2.73
5344	30605.38	80098.67	16.86
Throughput (KB/ms)	183.02	465.480	334466.828

The execution and working of the considered algorithms are shown in figure 1.

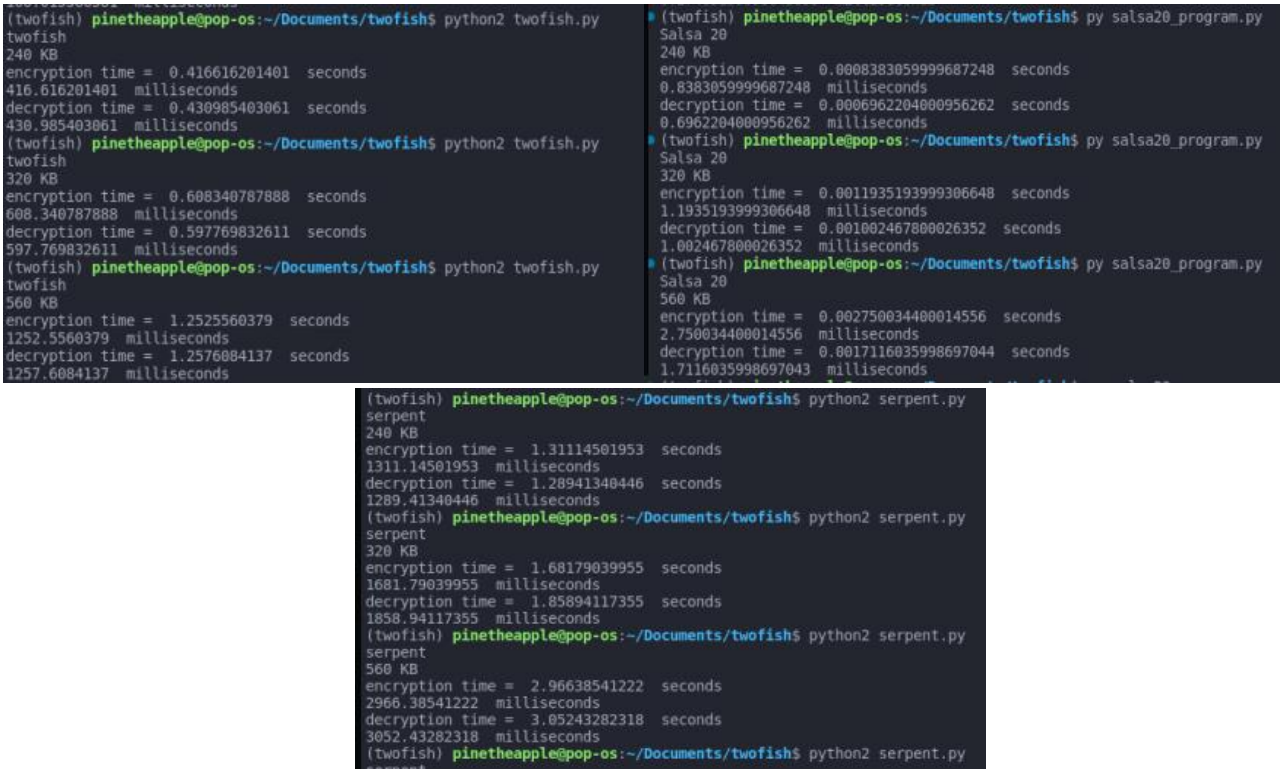

Figure 2: Execution of cryptographic algorithms

IV. CONCLUSION

After conducting the experiments and analyzing the results, it was found that Salsa20 outperformed the other two algorithms, Twofish and Serpent, by a significant margin. The results showed that Salsa20 had the fastest encryption and decryption times compared to the other algorithms, while also maintaining high levels of data security. Thus, it can be concluded that Salsa20 is a strong candidate for applications that require high-speed and secure encryption and decryption of data. However, it is important to note that the choice of algorithm also depends on specific application requirements and constraints.

REFERENCES

1. Bhagat, V., Kumar, S., Gupta, S. K., & Chaube, M. K. (2023). Lightweight cryptographic algorithms based on different model architectures: A systematic review and futuristic applications. *Concurrency and Computation: Practice and Experience*, *35*(1), e7425.
2. Gowda, N. C., & Srivastav, P. S. V. (2019). GPR: Steg Cryp (Encryption using steganography). *International Journal of Engineering and Advanced Technology (IJEAT)*, *8*.
3. Veena, H. N., & Gowda, N. C. (2018). Design and Implementation of Image Encryption using Chaos Theory. *Asian Journal of Engineering and Technology Innovation (AJETI)*, *208*.
4. Biham, E., Anderson, R., & Knudsen, L. (1998). Serpent: A new block cipher proposal. In *Fast Software Encryption: 5th International Workshop, FSE'98 Paris, France, March 23–25, 1998 Proceedings 5* (pp. 222-238). Springer Berlin Heidelberg.
5. Shah, T., Haq, T. U., & Farooq, G. (2020). Improved SERPENT algorithm: design to RGB image encryption implementation. *IEEE Access*, *8*, 52609-52621.
6. PN, K. K., Baruah, R., Dey, S., Pradeep, K. R., & Vahab, F. (2018). Real Time Air Quality Measurement Using Low Power Wide Area Technology. *Asian Journal of Engineering and Technology Innovation (AJETI)*, 153.
7. Anderson, R., Biham, E., Knudsen, L., & Technion, H. (1998, August). Serpent: A flexible block cipher with maximum assurance. In *The first AES candidate conference* (pp. 589-606).
8. Schneier, B., Kelsey, J., Whiting, D., Wagner, D., Hall, C., & Ferguson, N. (1998). Twofish: A 128-bit block cipher. *NIST AES Proposal*, *15*(1), 23-91.
9. Sawant, A. G., Nitnaware, V. N., Dengale, P., Garud, S., & Gandewar, A. (2019). Twofish algorithm for encryption and decryption. *Journal of Emerging Technologies and Innovative Research (JETIR)*, *6*(1).
10. Yerriswamy, T., & Murtugudde, G. (2021). An efficient algorithm for anomaly intrusion detection in a network. *Global Transitions Proceedings*, *2*(2), 255-260.
11. Gowda, N. C., Manvi, S. S., & Malakreddy, B. (2022, July). Blockchain-based Access Control Model with Privacy preservation in a Fog Computing Environment. In *2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)* (pp. 1-6). IEEE.
12. Manvi, S. S., & Gowda, N. C. (2019). Trust Management in Fog Computing: A Survey. In *Applying Integration Techniques and Methods in Distributed Systems and Technologies* (pp. 34-48). IGI global.
13. Standard, D. E. (1999). Data encryption standard. *Federal Information Processing Standards Publication*, *112*.
14. Hussain, S., Asif, M., Shah, T., Mahboob, A., & Eldin, S. M. (2023). Redesigning the Serpent Algorithm by PA-Loop and Its Image Encryption Application. *IEEE Access*, *11*, 29698-29710.
15. Lucks, S. (2002, June). The saturation attack—a bait for Twofish. In *Fast Software Encryption: 8th International Workshop, FSE 2001 Yokohama, Japan, April 2–4, 2001 Revised Papers* (pp. 1-15). Berlin, Heidelberg: Springer Berlin Heidelberg.

16. Ahmed, S. T., Ashwini, S., Divya, C., Shetty, M., Anderi, P., & Singh, A. K. (2018). A hybrid and optimized resource scheduling technique using map reduce for larger instruction sets. *International Journal of Engineering & Technology*, *7*(2.33), 843-846.
17. Shalini, L., Manvi, S. S., Gowda, N. C., & Manasa, K. N. (2022, June). Detection of Phishing Emails using Machine Learning and Deep Learning. In 2022 7th International Conference on Communication and Electronics Systems (ICCES) (pp. 1237-1243). IEEE.
18. Ambika, B. J., & Banga, M. K. (2021). A novel energy efficient routing algorithm for MPLS-MANET using fuzzy logic controller. *International Journal of Information and Computer Security*, *14*(1), 20-39.
19. Lara, E., Aguilar, L., García, J. A., & Sanchez, M. A. (2018). A lightweight cipher based on salsa20 for resource-constrained IoT devices. *Sensors*, *18*(10), 3326.
20. Yerriswamy, T., & Murtugudde, G. (2020, October). Study of Evolutionary Techniques in the field of Network Security. In *2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)* (pp. 594-598). IEEE.
21. Ahmed, S. S. T., Thanuja, K., Guptha, N. S., & Narasimha, S. (2016, January). Telemedicine approach for remote patient monitoring system using smart phones with an economical hardware kit. In *2016 international conference on computing technologies and intelligent data engineering (ICCTIDE'16)* (pp. 1-4). IEEE.
22. Raja, D. K., Kumar, G. H., Basha, S. M., & Ahmed, S. T. (2022). Recommendations based on integrated matrix time decomposition and clustering optimization. *International Journal of Performability Engineering*, *18*(4), 298.