PERFORMANCE MEASUREMENT OF A HYBRID ENCRYPTION SCHEME ON AN EMBEDDED SYSTEM

Luis Alejandro Vargas Oviedo Edwar Jacinto Gómez and Fernando Martínez Santa

Facultad de Ingeniería, Universidad Distrital Francisco Jose de Caldas, Bogotá, Colombia

ABSTRACT

In the last decades, information security has become a priority with exponential growth since technological developments must be supported so that their operation has reliable levels to store relevant data, and the levels of reliability are those required by the current environment. This is key to developing modern requirements that apply to data protection. It should be noted that the most common tool to encrypt information in code is AES (Advanced Encryption Standard), which is a block cipher algorithm that assigns binary keys and performs rounds of exchange to hide the message; on the other hand, is RSA (Rivest - Shamir - Adleman) which is an algorithm used for key assignment. Finally, the HASH key extension function is used for cryptographic analysis to verify the authenticity and origin of the data. In this research, a source code that combines the AES, RSA, and HASH encryption algorithms is designed to run on two hybrid acceleration units to compare the data processing in terms of time and reliability.

KEYWORDS

Information Security, Key Extension, System Embedded, Mixed Encryption Scheme, Hybrid Acceleration Unit.

1. INTRODUCTION

The exponential growth of technological processes in both industries and training centers [1] encourages the generation of reliable environments for handling information and controlling the security given to it [2] [3]. Symmetric block cipher algorithms such as AES [4][5], asymmetric encryption methods for a critical assignment such as RSA [6][7], and critical extensions using combinations of HASH functions [8][9] can be used in combination to increase the levels of reliability with which information of interest is being encrypted [10][11]. Algorithms such as those mentioned above seek to ensure that the control of the issued message is secure and that each code protects the part that corresponds to it [12 The AES encryption algorithm is used for the message and its encryption process in the generation and management of keys [13], The RSA public key algorithm is used for key allocation to obtain the private key, which usually takes longer if the key length increases [14], for the extension of dynamic keys and to increase the security layer, the HASH algorithm is used, considered to have high-security levels. [15].

The use of solutions with embedded components in information security processes is required to increase privacy blocks [16][17], Stand-Alone type, since elements containing high-capacity features improve performance and, therefore, the results in the environments where they are implemented, since decentralizing data processing tends to be more agile [18]. The fusion of technologies seeks to meet the current needs demanded by information security so that the stored data has additional support during data transmission through a communication channel [19]. This is intended to ensure that the contents are protected in a secure environment no matter where they are applied [20]. Using single-board hardware that can work with open-source code complemented by code that guarantees data reliability increases data processing and protection levels [21]; current research has found security support in electronic systems generating levels of data protection using AES applied to IoT devices [29]. the starting point for the mixture of algorithms to take importance [22]; as a result, it is aim that the messages sent through the corresponding channels generate an exponential behavior in terms of security so that the assignment, conversion, and encryption of keys support the privacy of the message issued [23][24]. Some industries that have ventured into the proper use of their internal information through technological tools promote innovation and competitiveness behaviors that enhance the added value of the same in the market [25].

In this article, we seek the development of an implementation that uses a cryptographic key in an embedded system [26][27], which performs the block ciphering of information using the AES algorithm [28], supported by the secure exchange of session keys using the RSA algorithm [29], along with the combination and key extensions applying combinations of HASH algorithms [30]. Thus, delivering a software-hardware tool, which has been tested in at least two devices with embedded Linux, in which a series of tests have been run to verify some metrics of interest, in order to have a solution that provides an additional level of information security to those currently available [31].

The scheme used for the execution of the research is as follows: In the second part, the methodology is presented; in this, the combined modeling of the encryption algorithms and together with the key extension that will be applied to the information to be encrypted is evidenced. The implementation is the third part; here, we find the assignment of the source code for each algorithm in its two versions, encrypted and decrypted. The results are considered as the fourth part of the work since the comparisons of the hybrid acceleration units are made in terms of information processing in each of the parts of the code; this is in order to determine the fifth and last part of the research, which are the conclusions where the best environment for data processing will be related.

2. METHODOLOGY

It is necessary to develop the research of an asymmetric algorithm under a hybrid system that uses public and private keys with an addition of key protection in its modeling to support the protection of information from an initial point to its final destination. In the following scheme (Figure 1), each of the algorithms applied to perform the encryption process will be identified:

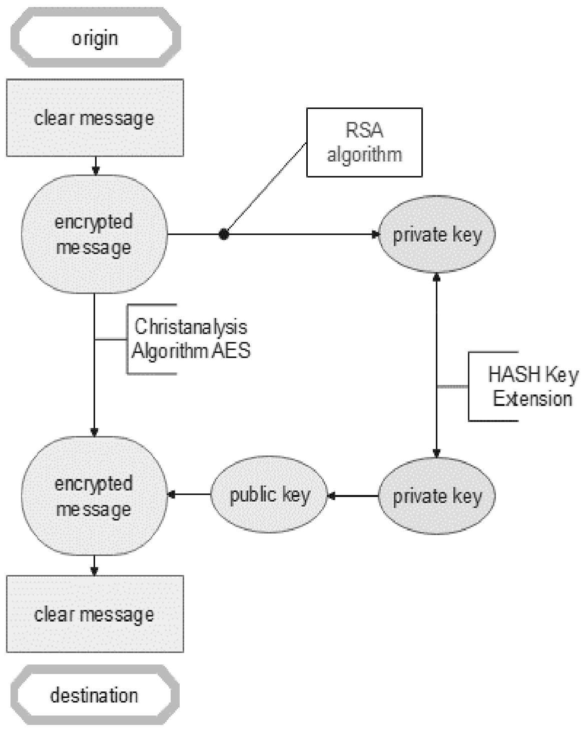


Figure 1. Mixed encryption scheme

In the mixed encryption scheme, the algorithms involved in the cryptographic process are identified; in the first instance, a Key extension is performed using a HASH algorithm as a base, which ensures that the size of the session key is equal to the size of the block cipher word and also increases the level of protection of the Key allocation to protect the information contained, such session key after being expanded under PKI standards, is shared using the RSA asymmetric algorithm, so that a secure exchange of the expanded key can take place, thus allowing the encryption and decryption processes to be carried out. On the other hand, the AES block cipher algorithm is used to perform the encryption process applied to the message in clear text, which, in the end, will be the information that will travel through the medium.

2.1. AES (Encryption Algorithm)

It is an algorithm that is used on the text, which is divided by blocks of 128/192/256 Bits to start its process of rounds that assign keys to it to increase the level of encryption. [7]. Subsequently, a key is generated for each block, and two matrices, the first called "State" and the second "Cipher Key", are combined to obtain the ciphertext after running the respective combination iterations [13]. It is used as a standard algorithm to encrypt information since the old methods lost effectiveness due to the exponential evolution of technology [28].

2.2. RSA Algorithm

RSA (Rivest - Shamir - Adleman) public key algorithm that gets its name from the surnames of its three creators is considered one of the most secure, and its application in the fusion of technologies has had significant contributions; it consists of the factorization of large numbers [6
For key generation, private keys must be assigned by the parties involved in the transmission of

the message. It should be noted that the longer the key is, the longer the data processing will take additional processing time. [14].

2.3. HASH Function - Key Extension

To solve the problems associated with information integrity, the HASH function is used in cryptographic analysis as a dynamic tool that increases the protection of the environments where it is applied. It is used to verify the authenticity and origin of the worked data [15].

The addition and combination of encryption algorithms increase the levels of reliability of the information handled. It is here where the programming, verification, and execution of the project take an added value since the fusion of technologies increases the security support in the transport channel of the emitted message. In this case, an application of the summary algorithms is made to perform an iterative calculation of the summary of the session key typed by the user to increase the security level of the key and that the key is not stored but rather this iterative summary.

3. IMPLEMENTATION

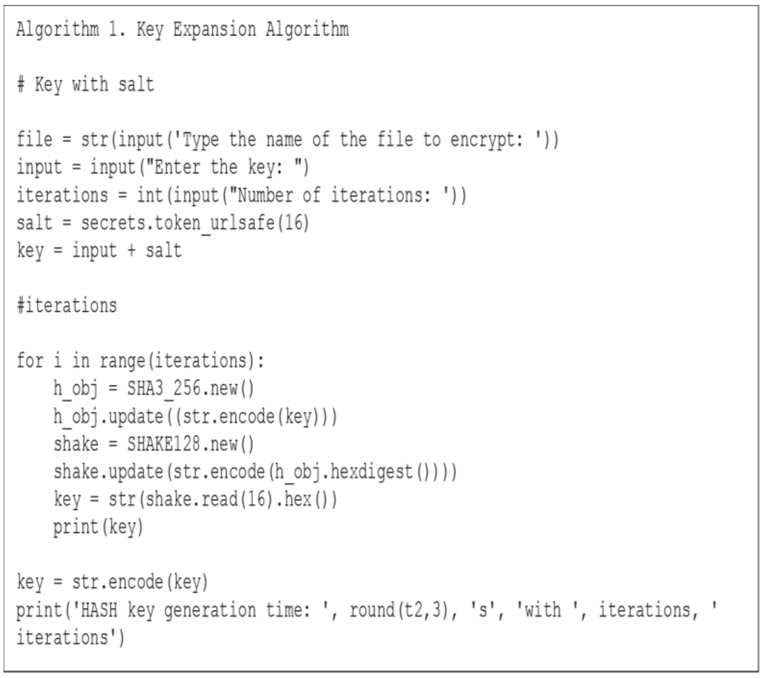
The first step is to create the source code where applying the algorithms above is the basis for the central part of the research. For this case, the Python software tool is used, adding the use of the pyCryptodome library, which performs the cryptographic operations of the AES, RAS, and HASH (essential expansion) encryption algorithms. As a second step, two hybrid acceleration units must act as cryptographic keys. The first used is the Raspberry Pi Zero Board Wifi Bluetooth 1ghz CPU 512m RAM, and the second is the NVIDIA Jetson Nano. A final analysis of the performance in data processing is sought when executing the code on each of the keys.

3.1. Encryption Algorithm

3.1.1. Key Expansion

In encryption processes where different algorithms are linked, the user who wishes to protect the information of interest must assign a key, after which a security addition is generated; the expansion algorithm allows the assigned keys to be stored and their use to have a secure environment. For this, a plain text capture is generated by the user and linked to a pseudo-random number called salt; these numbers correspond to a series of ASCII characters in a range corresponding to the Latin alphabet.

The "Digest" process is applied to the character string of the session key result together with the salt to create a key that meets the same dimensions as the block of information to be encrypted. The process is iterative to increase the level of security, thus increasing the level of trust since the use of the pseudo-random salt together with the "Digest" process generates randomness to shield the session key.

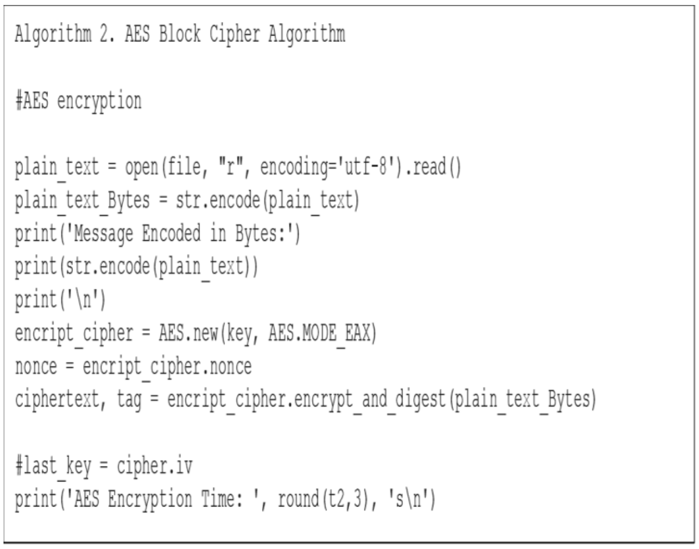


```
Algorithm 1. Key Expansion Algorithm

# Key with salt

file = str(input('Type the name of the file to encrypt: '))
input = input("Enter the key: ")
iterations = int(input("Number of iterations: '))
salt = secrets.token_urlsafe(16)
key = input + salt

#iterations

for i in range(iterations):
    h_obj = SHA3_256.new()
    h_obj.update((str.encode(key)))
    shake = SHAKE128.new()
    shake.update(str.encode(h_obj.hexdigest()))
    key = str(shake.read(16).hex())
    print(key)

key = str.encode(key)
print('HASH key generation time: ', round(t2,3), 's', 'with ', iterations, '
iterations')
```

Figure 2. Key Expansion Algorithm

3.1.2. Encryption using the AES algorithm

First of all, the information of interest to be encrypted is selected, to which the AES encryption algorithm is applied, which is responsible for dividing the text into blocks of a size equal to the number of bits of the cipher and the size of the key expansion, The use of the Cryptodome library in the aforementioned tasks generates a simple development to perform this task, it is only necessary to take into account that the data corresponding to the plain text must be encoded in a Bytes data type.

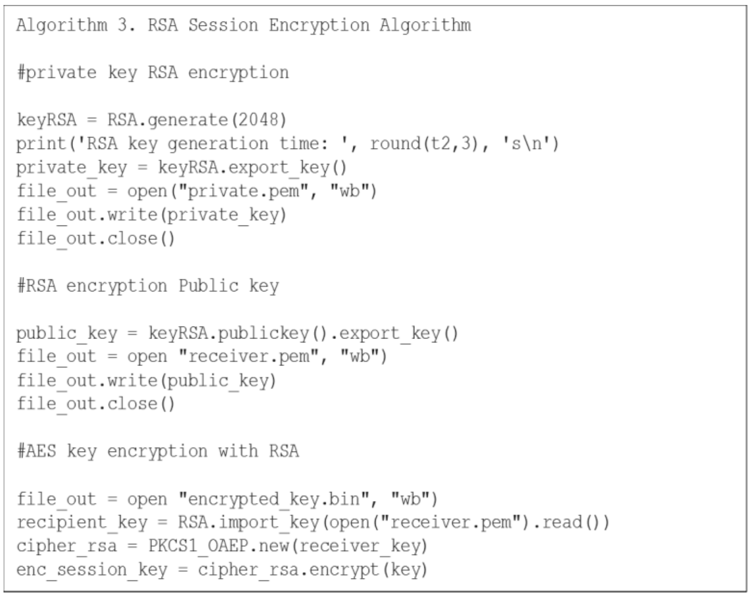


```
Algorithm 2. AES Block Cipher Algorithm

#AES encryption

plain_text = open(file, "r", encoding='utf-8').read()
plain_text_Bytes = str.encode(plain_text)
print('Message Encoded in Bytes:')
print(str.encode(plain_text))
print('\n')
encript_cipher = AES.new(key, AES.MODE_EAX)
nonce = encript_cipher.nonce
ciphertext, tag = encript_cipher.encrypt_and_digest(plain_text_Bytes)

#last_key = cipher.iv
print('AES Encryption Time: ', round(t2,3), 's\n')
```

Figure 3. AES Algorithm

3.1.3. Encryption using RSA algorithm

To increase security levels, both the sender and the receiver of the information must assign private keys. In this case, the source code associated with the RSA algorithm is applied to the recipient's public key in order to encrypt the session key and the final subkey. From this, the cryptographic analysis is executed to obtain the information from the recipient's private key.



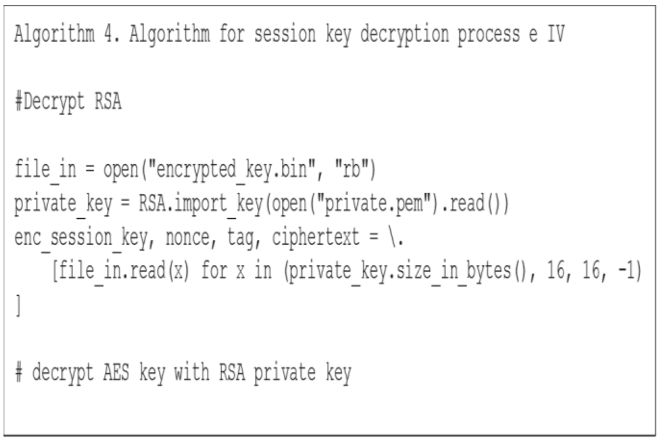
```
Algorithm 3. RSA Session Encryption Algorithm

#private key RSA encryption

keyRSA = RSA.generate(2048)
print('RSA key generation time: ', round(t2,3), 's\n')
private_key = keyRSA.export_key()
file_out = open("private.pem", "wb")
file_out.write(private_key)
file_out.close()

#RSA encryption Public key

public_key = keyRSA.publickey().export_key()
file_out = open "receiver.pem", "wb")
file_out.write(public_key)
file_out.close()

#AES key encryption with RSA

file_out = open "encrypted_key.bin", "wb")
recipient_key = RSA.import_key(open("receiver.pem").read())
cipher_rsa = PKCS1_OAEP.new(receiver_key)
enc_session_key = cipher_rsa.encrypt(key)
```

Figure 4. RSA Algorithm

3.2. Decryption Algorithm

3.2.1. Decryption RSA algorithm

For this part of the code, we proceed to decrypt the session key and the last encrypted subkey from the recipient's private key. The application of a simple algorithm to take the encrypted information and run a process of exponentiation and modular operation will branch the processing of the keys and thus the AES decryption process can be performed.

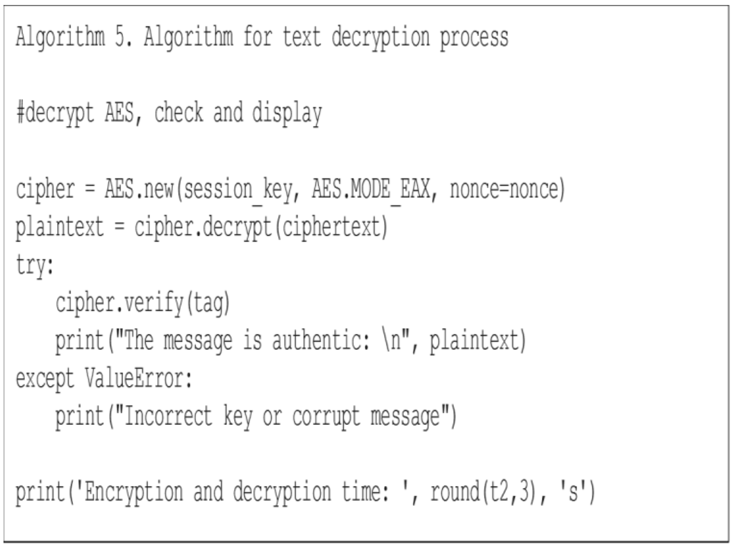


```
Algorithm 4. Algorithm for session key decryption process e IV

#Decrypt RSA

file_in = open("encrypted_key.bin", "rb")
private_key = RSA.import_key(open("private.pem").read())
enc_session_key, nonce, tag, ciphertext = \.
    [file_in.read(x) for x in (private_key.size_in_bytes(), 16, 16, -1)
]

# decrypt AES key with RSA private key
```

Figure 5. RSA Algorithm – Decryption

3.2.2. Decryption of the AES algorithm

Finally, the AES algorithm is decrypted by receiving the information with the secret text. The session key, the last decrypted subkey, and the cipher text are received. After applying this process, the text of the clear message is obtained.



```
Algorithm 5. Algorithm for text decryption process

#decrypt AES, check and display

cipher = AES.new(session_key, AES.MODE_EAX, nonce=nonce)
plaintext = cipher.decrypt(ciphertext)
try:
    cipher.verify(tag)
    print("The message is authentic: \n", plaintext)
except ValueError:
    print("Incorrect key or corrupt message")

print('Encryption and decryption time: ', round(t2,3), 's')
```

Figure 6. AES Algorithm – Decryption

4. RESULTS

After the creation of the source code and in order to measure the data processing in terms of time, a comparison of the modeling of each of the algorithms applied to the development of the research with their respective encryption and decryption processes is made. The tests were executed in two hybrid acceleration units; one of them is the Raspberry pi in its versions 1B and 3B+, and the other test was executed in the NVIDIA Jetson Nano.

The encrypted information has a size of 23 KB; for the extension of the key by means of the Hash algorithm, 100 iterations were executed. After the password extension, the AES algorithms were run for the block message and the RSA algorithm for the key assignment. The following table shows the times obtained in each of the cryptographic keys, segmented into the algorithms applied, and illustrates the total time of the execution of the encryption process.

Table 1. Data processing times

cryptographic key	HASH(s)	AES(s)	RSA(s)	ENC/DEC(s)	TOTAL(s)
Raspberry pi 1B	0,225	0,042	27,988	3,867	32,747
Raspberry pi 3B+	0,036	0,01	10,12	0,568	10,806
Jetson NANO	0,037	0,004	2,519	0,339	2,95

Figure 7 shows the total time comparison of the encryption and decryption process of the hybrid acceleration units, where optimal data processing tends to be minimal in one of the three applications since, in the tests executed, the Jetson nano has an efficient performance compared to the others. This key presents optimal performance in computational use due to implementing parallel neural networks that allow faster data processing.

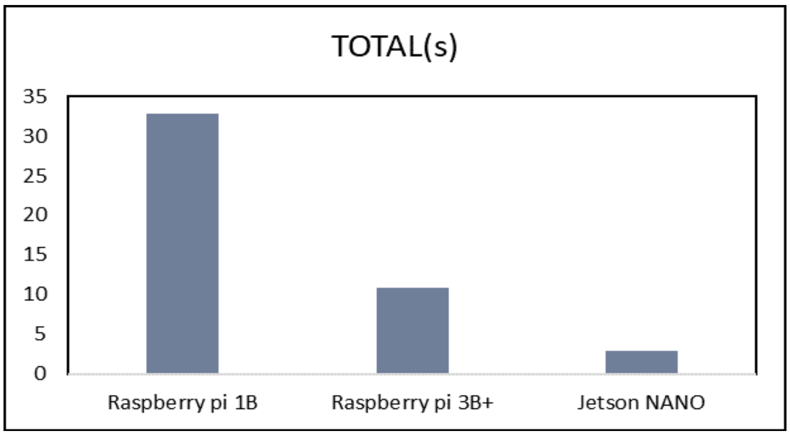


Figure 7. Total encryption times

5. Conclusions

This research reveals that the mixture of encryption algorithms and hardware devices with parallel architecture improves the functional performance minimally in a double and consequently enhances information protection in comparison with devices with classical architecture. Encryption segmentation is divided into parts that focus on a particular fragment of the process: the message, the assignment of keys, or the extension of passwords.

Upon receiving the test results, it was found that data processing is more efficient with the Jetson Nano hybrid acceleration unit, as its modeling and information processing are 90.99% faster than the Raspberry Pi 1B and 72.70% more efficient than the Raspberry Pi 3B+.

The fusion of technological tools allows the development of applications that promote the good use of information regardless of the environment where it is being used; the inclusion of tangible devices in these information protection techniques broadens the field of work for the generation of new knowledge.

6. Future Works

The research group will develop solutions using a hybrid encryption system with steganography, applying public key cryptographic techniques to add levels of blocking to information transmitted over hardware devices.

7. Real-World Applications

Cryptography is a method of reference in the protection of information; its applications in corporate environments where the HSM (hardware security module) is used: card payment systems, m-commerce, access control, and trusted platform modules; in order to protect sensitive information have increased in recent years; on the other hand, one of the sectors that have most ventured into data protection is the financial sector, its mission to ensure the protection of information and capital motivates them to adopt these methodologies and finally the application in academic environments is essential for the generation of new growth environments.

REFERENCE

[1] Holman, M. A., Martínez, F., & Edwar, J. G. (2022). Implementation of Password Hashing on Embedded Systems with Cryptographic Acceleration Unit. *International Journal of Advanced Computer Science and Applications*, *13*(2).

[2] Alhabeeb, O. H., Fauzi, F., & Sulaiman, R. (2021). A Review of Modern DNA-based Steganography Approaches. *International Journal of Advanced Computer Science and Applications*, *12*(10).

[3] Rana, S., Mondal, M. R. H., & Parvez, A. S. (2021). A new key generation technique based on neural networks for lightweight block ciphers. *International Journal of Advanced Computer Science and Applications*, *12*(6).

[4] Do Xuan, C. (2021). A Proposal to Improve the Bit Plane Steganography based on the Complexity Calculation Technique. *International Journal of Advanced Computer Science and Applications*, *12*(6).

[5] Fadhil, M. S., Farhan, A. K., & Fadhil, M. N. (2021). A lightweight AES Algorithm Implementation for Secure IoT Environment. *Iraqi Journal of Science*, 2759-2770.

[6] Abroshan, H. (2021). A hybrid encryption solution to improve cloud computing security using symmetric and asymmetric cryptography algorithms. *International Journal of Advanced Computer Science and Applications*, *12*(6).

[7] Alfaleh, F., & Elkhediri, S. (2021). Efficient Security Solutions for IoT Devices. *International Journal of Advanced Computer Science and Applications*, *12*(4).

[8] Ahmed, H. (2022). A Review of Hash Function Types and their Applications. *Wasit Journal of Computer and Mathematics Sciences*, *1*(3), 116-135.

[9] Pantoja, N. D., Donado, S. A., Villalba, K. M., & Moreno, S. X. (2021). Comportamiento del algoritmo RSA en diferentes en diferentes arquitecturas computacionales. Revista Ibérica de Sistemas e Tecnologías de Información, (E41), 358-369

[10] Krishna, P. G., & Muthuluru, N. (2021). Feistel Network Assisted Dynamic Keying based SPN Lightweight Encryption for IoT Security. *International Journal of Advanced Computer Science and Applications*, *12*(6).

[11] Tushar, A. S., & Mishra, A. (2021). 3 "Cryptographic Algorithm For Enhancing Data Security: A Theoretical Approach.". *International Journal of Engineering Research & Technology*, *10*(03).

[12] Patil, A. S., & Sundari, G. (2021). Pixel Value Difference based Face Recognition for Mitigation of Secret Message Detection. *International Journal of Advanced Computer Science and Applications*, *12*(2).

[13] Clavijo, A. M., & Chacón, J. A. (2023). Prototipo de cifrado híbrido combinando los métodos de encriptación AES y RSA. Revista Vínculos, 20(1).

[14] Guerrero, R. R., Vanegas, C. A., & Montiel, G. G. C. (2024). Análisis de Eficiencia del Algoritmo RSA en Entornos Móviles de Bajo Desempeño Computacional. *XIKUA Boletín Científico de la Escuela Superior de Tlahuelilpan*, *12*(Especial), 13-19.

[15] Srivastava, V., Baksi, A., & Debnath, S. K. (2023). An overview of hash based signatures. *Cryptology ePrint Archive*.

[16] Hadjadj, M. A., Sadoudi, S., Azzaz, M. S., Bendecheche, H., & Kaibou, R. (2022). A new hardware architecture of lightweight and efficient real-time video chaos-based encryption algorithm. *Journal of Real-Time Image Processing*, 1-14.

[17] Krishna, B. M., Santhosh, C., Suman, S., & Shireen, S. S. (2022). Evolvable hardware-based data security system using image steganography through dynamic partial reconfiguration. *Journal of Circuits, Systems and Computers*, *31*(01), 2250014.

[18] Liu, C., Zhang, Y., Xu, J., Zhao, J., & Xiang, S. (2022). Ensuring the Security and Performance of IoT Communication by Improving Encryption and Decryption With the Lightweight Cipher uBlock. *IEEE Systems Journal*.

[19] Jian, M. S., & Wu, J. M. T. (2021). Hybrid Internet of Things (IoT) data transmission security corresponding to device verification. *Journal of Ambient Intelligence and Humanized Computing*, 1-10.

[20] Ordoñez, H., Azadobay, J., Morales, M., & Montenegro, C. (2022). Automatización Web del Proceso de Votación de las Elecciones de la EPN Utilizando Esquema de Seguridad de Firma Ciega. Latin American Journal of Computing, 9(2), 66-79.

[21] Gamess, E., & Hernandez, S. (2022). Performance Evaluation of Different Raspberry Pi Models for a Broad Spectrum of Interests. *International Journal of Advanced Computer Science and Applications*, *13*(2).

[22] Al-Kasasbeh, B. (2022). A Novel Secure Transposition Cipher Technique using Arbitrary Zigzag Patterns. *International Journal of Advanced Computer Science and Applications*, *13*(1).

[23] Mishrar, M. S., & Mahalakshmi, J. (2022). A BREIF STUDY ON VARIOUS CRYPTOGRAPIC ALGORITHMS FOR DATA SECURITY. *IJRAR-International Journal of Research and Analytical Reviews (IJRAR)*, *9*(2), 288-293.

[24] Kaur, S., Shinde, P., Joshi, J., & Dakhare, B. S. (2021). Analysis of Security Algorithms in Cloud Environment. *International Journal of Research in Engineering, Science and Management*, *4*(5), 33-37.

[25] Kumar, K. K. (2021). Private Attached Network Cloud Storage Using IOT. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, *12*(9), 53-55.

[26] Marianetti, O., Chediack, E., & Fontana, D. (2023). Implementación segura de sistemas de IoT. *Memorias de las JAIIO*, *9*(11), 11-14.

[27] Sarmiento, C., Aldaya, A. C., Santiago, S. Ã., Junior, E. I., Pimenta, T. C., & Brumley, B. B. (2023). Contribuciones a la implementaciÃ³n de sistemas electrÃ³nicos digitales embebidos sobre hardware reconfigurable. *Anales de la Academia de Ciencias de Cuba*, *13*(4), 1482.

[28] Vaz, YS, Mattos, JC y Soares, RI (noviembre de 2023). AES optimizado para su uso en aplicaciones de IoT. En *Actas Ampliadas del XIII Simposio Brasileño de Ingeniería de Sistemas Informáticos* (págs. 31-36). SBC.

[29] Donato¹, R. A. C., Silva¹, H. D., Cavalcante¹, J. V. F., & Araújo¹, F. C. (2023). Aplicação de Criptografia E2EE no Compartilhamento de Dados em Redes com o Algoritmo RSA.

[30] Wang, F., Chen, Z., Wang, G., Song, Y. y Liu, H. (2024). Codificación hash espacio-temporal enmascarada para reconstrucción dinámica de escenas eficiente. *Avances en sistemas de procesamiento de información neuronal , 36* .

[31] Chmielewski, Ł., & Weissbart, L. (2021, June). On reverse engineering neural network implementation on GPU. In *International Conference on Applied Cryptography and Network Security* (pp. 96-113). Springer, Cham.

AUTHORS

Luis Alejandro Vargas Oviedo is a production engineer from the Universidad Distrital, working in Colombia. He is a student of the Master in industrial engineering with emphasis in technological projects. his areas of research are cryptography, process optimization in flexible manufacturing cells.

Edwar Jacinto Gómez is an electronic control and instrumentation engineer working in Colombia. He is an associate professor at the district university since 2008 and has a master's degree in information science and communications. His research areas are cryptography, advanced digital, image processing and robotics where he has publications on the subject.

Fernando Martinez Santa is an Electronic Control and Instrumentation Engineer working in Colombia. He is an associate professor at the Universidad Distrital and has a master's degree in electronic and computer engineering. His research areas are cryptography, microcontroller architecture and robotics.