Generation and Evaluation of a New Time-Dependent Dynamic S-Box Algorithm for AES Block Cipher Cryptosystems

Hayder T. Assafli 1,* and Ivan A. Hashim 2

1 Department of Electrical Engineering, University of Technology, Iraq
E-mail: 140030@uotechnology.edu.iq
2 Department of Electrical Engineering, University of Technology, Iraq
E-mail: 30095@uotechnology.edu.iq

* Corresponding author: Department of Electrical Engineering, University of Technology, Iraq. Tel: +9647901772796.

Abstract. Cryptographic communication systems depend extensively on the Advanced Encryption Standard (AES) block cipher algorithm. The encryption strength in the AES algorithm is derived from the nonlinearity introduced by the application of s-boxes inside the encryption process. Today, most s-boxes are either static in which the s-box content does not change during the encryption process or dynamic which continuously changes depending on the encryption key. Two major problems exist in such systems that increase the risk of unauthorized decryption. First, the encryption of the same data yields the same cipher key repeatedly. Secondly, the key-dependent s-boxes require sharing the encryption key each time a new s-box is generated. In this paper, a dynamic s-box is introduced that constantly changes independent of the encryption key. The new s-box depends on the epoch timestamp that exists in every digital system as well as transmitted with digital satellite communication systems. The generation of the nonlinear s-box occurs inside the transmitter and receiver identically. The main strength of the proposed approach is that the ciphertext changes while keeping the encryption key constant ensuring different encryption results for the same data. This paper also investigates and analyzes the strength and quality of the new s-box using the avalanche effect method. The results showed that different ciphertext is produced after every encryption process. Moreover, the modified dynamic s-box reached a high avalanche effect exceeding the Strict Avalanche Criterion (SAC).

Keywords: Encryption; Advanced Encryption Standard; S-box, Avalanche effect; and Epoch time.

1. Introduction
Block cipher cryptographic algorithms are becoming more susceptible to attack by hackers with the
developments in computation power and speed. Therefore, high-security systems are required to protect important and valuable information from the access of unauthorized users (Park et al., 2018). Cryptographic algorithms are among the most effective methods in which the desired information is hidden inside cyphered text. Only users with the encryption key have access to the information by the decryption process. When the encryption and decryption keys are identical, the cryptographic algorithm is called symmetric as in Advanced Encryption Standard (AES) and Data Encryption Standard (DES). On the contrary, using a different key for decrypting data is called asymmetric method such as Rivest-Shamir-Adleman algorithm (RSA) (Kumar & Tiwari, 2012). The cryptographic algorithms are required to include nonlinearity in the encryption process. Several encryption methods depend on s-boxes for providing strength to the diffusion process such as Data Encryption Standard (DES), International Data Encryption Algorithm (IDEA) (Al-Dweik et al., 2019), and PRESENT algorithms (Jithendra & Shahana, 2019). Advanced Encryption Standard became a major focus for several studies in the previous decade. The AES is used as a major encryption process in several applications including media such as images (Alsaffar et al., 2020; Singh et al., 2019) and videos (Adiguna & Hendrawan, 2017). Researches have been going on to improve the performance of AES encryption from several criteria such as strength, speed, and power consumption (Ueno et al., 2020). This research mainly focuses on introducing a new dynamic key independent s-box for increasing the strength of the AES algorithm (Patil et al., 2016).

2. Related Work

Using dynamic s-box for substitution in AES or other forms of cryptographic algorithms is not new to the field of cryptography. Indrajit Das et al. (Das et al., 2012) suggested that the safety of the AES algorithm can be increased by generating random dynamic s-boxes depending on irreducible polynomial. However, the polynomial must be sent to the receiver with the encryption key. Moreover, the polynomial occupies extra bandwidth and is exposed to cyber-attacks.

Lamiaa A. Elazm et al. (Elazm et al., 2010) introduced a high speed with a low power consumption s-box for the AES algorithm. They depended on the composite Galois field for the substitution box architecture. The size and delay of the s-box circuit is reduced compared with the standard complementary metal-oxide-semiconductor (CMOS) based systems. On the other hand, Manjula G. and Mohan H. S. (Manjula & Mohan, 2018) investigated the hash function for generating dynamic s-boxes for AES encryption. Their experiment results showed that the time of the encryption process is increased with the AES encryption strength and the time of the decryption process is decreased. Their method requires additional time for the dynamic s-box generation.

E. Siva Ganesh et. al. (Ganesh et al., 2012) proposed a method for generating dynamic s-boxes depending on the biometric scheme. They showed that the nonlinearity of the suggested s-box is increasing resulted in a higher strength AES system. The speed of their method qualifies their algorithm for encryption applications. However, the biometric scheme must be represented in both the encryption and decryption process. Kamsiah Mohamed et al. (Mohamed et al., 2014) and Hui Shi et al. (Shi et al., 2011) studied the nonlinearity effect of the s-box in the AES encryption process. Since the s-box is the only step that introduces nonlinearity effect to the process, the effect must be high enough not to be hacked by attackers. They both concluded the avalanche effect is the most important criterion that must be investigated to analyze the strength of the s-boxes.

Pham Anh Hung et al. (Hung et al., 2013) used two different chaotic maps for generating dynamic s-boxes for usage in encryption. Their suggested method adds more shuffling to the s-box matrix elements increasing the overall strength of the AES encryption method. Do Quang Huy (Huy et al., 2019) used Galois field (GF) instead of the standard lookup tables for generating dynamic s-boxes. They proposed that the usage of s-box and inverse s-box on the same hardware chips makes it suitable for applications such as wireless local area networks. Julia Juremi et al. (Juremi et al., 2012) rotated the static s-box of the basic AES encryption structure depending on the encryption key. This method does not require sharing additional information between the sender and receiver. However, any change in the encryption key must be updated in the decryption key in order to continue the AES operation.

Kazys Kazlaukas and Jaunius Kazlaukas (Kazlaukas & Kazlaukas, 2009) suggested and analyzed a random s-box generator depending on the encryption key. They proposed that the usage of random
operation in the generation process results in higher strength AES algorithm without prediction of the generated s-box. Lijuan Li and Shuguo Li (Li & Li, 2017) used a shared s-box in the encryption-decryption process instead of the inverse s-box. The sharing process resulted in higher throughput according to their hardware implementation on an FPGA device. On the other hand, M. Kermani and A. Masoleh introduced a fault detection method for s-boxes in AES algorithms. Their approach is based on the low-cost composite field implementations. Another research implemented the s-box on hardware is done by Shreenivas Pai N (Shreenivas Pai et al., 2017). They applied Rijndael s-box depending on the combinational logic circuit in AES for application-specific integrated circuits. The reduced the occupancy area and the consumption of power in their suggested s-box module.

Previous researches focused on generating dynamic s-boxes depending on resources that must be present in both the sender and receiver. Any modification on one side requires an update on the other side which represents a major source of risk of encryption key exposure. The solution for such risk is to provide a nonce number for both senders and receivers such that the digits mapping process is concealed inside the encryption algorithm. This process keeps the encryption key hidden and achieves all previous research goals.

3. Advanced Encryption Standard (AES)

In 2001, the National Institute of Standards and Technology (NIST) selected the AES algorithm to succeed the older Data Encryption Standard (DES) which became weak and unable to stand hacking attacks [ref]. The name Rijndael algorithm is also applied to the AES algorithm because of the developers J. Daemen and V. Rijmen (Daemen & Rijmen, 2002). AES encryption and decryption method is a block cipher algorithm. In this type, the stream of data is divided into 128 bits blocks, and then each block is encrypted with 128, 192, or 256-bit key depending on the desired application requirement. The data blocks are named states and are arranged as 4x4 matrices. The blocks go through serial data modification processes resulting in cyphered blocks that are completely not understandable to the public. The flow chart represents the complete encryption process is shown in Figure 1 (Huy et al., 2019). The four basic operations used in AES encryption are adding a round key, byte shifting, mixing columns, and substituting bytes.

![Flow chart of AES encryption](image-url)

**Figure 1.** Flow chart of AES encryption
Substitute bytes:
A high nonlinearity is introduced to the encryption process by substituting bytes in the s-box. This step adds more confusion to the encryption operation.

Shift rows:
The rows of the 4x4 blocks are shifted depending on the position of each row. The first row is unaltered and the rest rows are shifted in cycles according to their positions. This process increases the scrambling of the encrypted data.

Mix columns:
A fixed matrix is used to mix the data block columns by multiplication. The matrix must satisfy the Galois Field (8th power of 2).

Add round key:
The working data block and the round key of each cycle are XORed as a simple addition process (Chennam et al., 2017).

The addition process is the first step in the encryption operation in which the encryption key is XORed with the first message block. Then, a loop starts including all the four operations mentioned above. The number of cycles depends on the size of the encryption key used to protect data. 10, 12, or 14 cycles are used for 128, 192, or 256 respectively (Lee et al., 2010). The final step is adding the round key again to the encrypted block resulting in the final cyphered text (Vaidehi & Rabi, 2014). Inversing the encryption process exactly results in the decrypted text with the same encryption key. Advanced Encryption Standard can be operated in six different modes depending on the requirement of the application. All six modes share the four encryption processes except slight differences in arrangement or technique. Electronic Code Book mode is the simplest operation mode in which a single key and a static s-box are used without initialization vector (Abdelrahman et al., 2018).

3.1 The S-Box

The substitution process in the AES encryption system performs nonlinear alterations on the encryption of data. It fulfills one of the main requirements for data security described by Shannon by applying the confusion property inside the algorithm (Shannon, 1998). Shannon suggested that repeating transposition with substitution leads to secure ciphers. Substitution permutation networks (SPN) were used in early cryptographic systems for combining permutation and substitution circuits. On the other hand, modern cryptographic systems must contain at least one nonlinear process for strengthening the encryption process (Hosseinkhani, 2012). As mentioned earlier, the s-box is the most essential step in the AES algorithm which requires the same length for both the message and the key. However, the highest delay in the AES algorithm is the s-box substitution process (Shivkumar & Umamaheswari, 2011). There are two types of s-boxes: static and dynamic. Static s-boxes have fixed contents during each round in the encryption process. Oppositely, the inverse s-box is fixed during the decryption process and it is the inverse transformation of the original s-box. Table 1 shows the static S-box used in the AES algorithm (Hosseinkhani, 2012). The first row and column represents the input to the s-box. When the two input numbers intersect, the addressed number is submitted as output. The static s-box is built as a simple logic circuit. The second type of the s-boxes is a dynamic s-box. In this type, the s-box is generated instantaneously with the key or depending on other sources of information. Generating s-box consumes time resulting in a slower encryption process. Nevertheless, the cipher key generated by dynamic s-boxes is stronger and have higher security than static s-boxes (Schneier & Sutherland, 1995). Special considerations must be taken for s-box when designing the AES algorithm. The most important consideration is the avalanche effect (Shi et al., 2011).
3.2 The Avalanche Effect

The avalanche effect is one of the important criteria that evaluates block cipher algorithms and describes the cryptographic function property. It is used to evaluate the effect of change in the output bits with respect to the input bits. In other words, it measures the flipping of output bits when one bit is flipped in the input. Strong cryptographic algorithms have a radical change in the output cyphered text due to a small change in the input data or encryption key. Failing to fulfill the requirement of the avalanche effect at a specific degree results in insignificant randomization and leads to the prediction of the encrypted information without the encryption key. The Strict Avalanche Criterion (SAC) was introduced by Webster and Tavares in 1985. This criterion states that at least 50% of the output bits are required to flip with the change of one bit in the input. The SAC became a standard for examining and evaluating cryptographic algorithms against cyber-attacks (Webster & Tavares, 1986).

4. Methodology

We propose a new method for modifying the AES-ECB encryption algorithm to increase the encryption strength. This method depends on the usage of a dynamic s-box instead of static s-box. Previously proposed algorithms generate the s-box dynamically with the encryption key. However, the proposed approach generates the s-box with each encryption process without any modification in the encryption key. The dynamic s-box generation depends on the Epoch time or Unix time on every encryption cycle. The Unix time has a unique one time value at every moment. This value is synchronized globally with all digital systems including mobile phones, computers, and global positioning system satellites. So, systems can generate their dynamic s-boxes independently without transmitting any information related to the encryption process. In other words, the encryption key is kept securely inside systems while the s-boxes are changed continuously. Ten digits are used to represent the UNIX time. These digits are fed into a dynamic s-box generator after being multiplied by a constant each called shift number. The proposed dynamic s-box generator accepts up to 16 double numbers. This gives the algorithm six spare digits that can be used for additional applications such as addressing. The
distribution of the epoch digits alongside with the shift number introduces extra strength to the AES-ECB encryption algorithm. Only the encryptor and the decryptor are supplied with the correct mechanism and information for the cryptographic process. Figure 2 demonstrates the suggested encryption method for higher security AES.

![Dynamic s-box generation with Unix time.](image)

Each digit of the Unix time is increasing continuously and the speed of change of each digit is related to a certain period. Minutes, hours, months, or years can be associated with a specific digit. This feature gives the encryption algorithm the ability of limited time operation. Thus, an expiration date can be embedded inside a system such that a system can decrypt data only for a specific limited time. Different encryption algorithms depend on the s-boxes for introducing the nonlinearity effect in the encryption process. The time dependent s-box can be modified to be used in other block cipher algorithms making the hacking process impossible even when the encryption key is known.

The simulation setup starts with a simple input text “1234567090abcde” which is converted into hexadecimal before encryption. Any combination of text sources can be used as a source for input text. Then, the hexadecimal result is entered to the AES encryption algorithm as a text source. Meanwhile, a dynamic s-box generator with internal epoch time generates an s-box at each request. The s-box is used in the substitution step. The rest of the processes are the same AES encryption steps. Using a new s-box each time ensures the generation of a new ciphertext after each execution of the encryption algorithm.

### 5. Results and Evaluation

To evaluate the dynamic s-box for AES encryption, several tests have been conducted using Matlab 2020a. There is no specific toolbox for algorithm testing in Matlab. Therefore, a software program is written, executed, tested and verified for the Avalanche effect testing. The AES algorithm is executed repeatedly to encrypt a simple text. The execution depended on different epoch time on each run. Different ciphertext is produced from the encryption process depending on different dynamic s-boxes. The resulted ciphertext is passed through the avalanche effect test. Results strongly suggested that encrypting with the suggested dynamic s-box leads to a completely different ciphertext each time. Moreover, the avalanche effect satisfies the SAC by maintaining higher than 0.5 most of the time. Table 2 shows the resulted ciphertext from the same message text for the first ten runs. Figure 3 illustrates the change in the avalanche effect during the first ten runs.
| Message Type | Hexadecimal code | Avalanche Effect | Unix Time |
|--------------|------------------|------------------|-----------|
| Encoded Message | 3123334353637303930636646560 | ---- | ---- |
| Encryption Key | 87006B08B4C29AE00BB7EC871619DD6 | ---- | ---- |
| Encryption 1 | A7000F0B0AED86864B33AA03121DDBA | 0.53906 | 1593355648781 |
| Encryption 2 | CF82B60BA292EAE80D37EA83D6D91D6 | 0.52343 | 1593355648785 |
| Encryption 3 | E3046F4CB08EF6C284F31EA47565FDDE | 0.51562 | 1593355648787 |
| Encryption 4 | C5606F0B809EE2924D77EA01101E9DA | 0.53906 | 1593355648790 |
| Encryption 5 | CB4C824BB59EE60BB7CE03961D1DA | 0.46875 | 1593355648792 |
| Encryption 6 | C702768B8AE9EE60B7EE075619DD6 | 0.50781 | 1593355648795 |
| Encryption 7 | A7402F6CD48EF6C80975AA05521D99A | 0.46875 | 1593355648797 |
| Encryption 8 | 8724476B4C2B2C6AC9F12CC59619DF6 | 0.49218 | 1593355648799 |
| Encryption 9 | C32CB300BAE20D2E2ACB77ECC59259D96 | 0.52343 | 1593355648801 |
| Encryption 10 | 87006B08B4C29AE00BB7EC871619DD6 | 0.50781 | 1593355648803 |

**Table 2. AES encryption results with dynamic s-box**

**Figure 3. The Avalanche effect for a discrete time**

Executing the algorithm continuously yields continuous fluctuations of the Avalanche effect as shown in Figure 4. The statistics for the observed fluctuations is shown in Table 3. It can be noted that the usage of the dynamic s-box satisfies the Strict Avalanche Criterion (SAC) for 75% of the time with a maximum of 0.63 exceeding the preferred SAC which is 0.5. It is worthwhile to mention that using constant s-box results in constant avalanche effect during the encryption process. The avalanche effect of static s-box depends on both the encryption key and input text. This raises the possibility of reducing the avalanche effect into undesired value (below 0.5). On the other hand, The suggested method does not require using different binary codes for mapping the input text to acquire such high avalanche effect.
as in (Dewangan et al., 2012). Moreover, the suggested approach stays higher than 0.5 most of the time exceeding the original static s-box which stays fairly at 0.5 (Shi et al., 2011). Furthermore, replacing the static s-box with a dynamic s-box in previous works did not achieve such high results and the avalanche effect remained close to 0.5 (Nejad et al., 2014; Tran et al., 2009; Bhoge & Chatur, 2014). Averaging the Avalanche effect during the runtime shows that it is also above 0.5 satisfying SAC. In other words, every encryption process uses the same but generates different outputs for the same input text. This increases the encryption security and does not require sharing any modification in the encryption key as in (Arrag et al., 2013; D’souza & Panchal, 2017; Hung et al., 2013) and (Nejad et al., 2014).

| Encryption Runs | Minimum | Maximum | Mean | Standard Deviation | Satisfies SAC | Dissatisfies SAC |
|-----------------|---------|---------|------|--------------------|--------------|------------------|
| 10000           | 0.4063  | 0.6328  | 0.5129| 0.0263             | 75.51%       | 24.49%           |

![Figure 4. The Avalanche effect for continuous-time](image)

### Table 3. Statistics of the Avalanche effect

6. Conclusion and Future Work

An enhanced mode of AES-ECB having high security is presented in this paper. The suggested algorithm is different from the original AES method such that it depends on the generation of dynamic s-boxes instead of a single static s-box. The creation of the dynamic s-box depends on the Unix time as a source of nonrepeatable number. The algorithm is tested and analyzed using the Avalanche effect in both discreet and continuous time. The results clearly showed that using the proposed algorithm generates a new ciphertext at each execution decreasing the risk of leaking the encryption key. Moreover, the achieved results satisfied the Strict Avalanche Criterion more than 70% of the execution time. Using the algorithm in future applications makes the communication process safer and more vulnerable to attacks. The future work is to focus on the implementation of the proposed algorithm in FPGAs for encrypting database systems.

### References

[1] Abdelrahman, A. A., Fouad, M. M., Dahshan, H., & Mousa, A. M. (2018). High performance CUDA AES implementation: A quantitative performance analysis approach. Proceedings of Computing Conference 2017, 2018-Janua(July), 1077–1085. https://doi.org/10.1109/SAI.2017.8252225
[2] Adiguna, T., & Hendrawan. (2017). Secure H.264 Video Coding using AES/CFB/PKCS5 padding encryption on various video frames (I, P, B). Proceeding of 2016 10th International Conference on Telecommunication Systems Services and Applications, TSSA 2016: Special Issue in Radar Technology, 5–9. https://doi.org/10.1109/TSSA.2016.7871104

[3] Al-Dweik, A. Y., Hussain, I., Saleh, M. S., & Mustafa, M. T. (2019). A Novel Method to Generate Key-Dependent S-Boxes with Identical Algebraic Properties. 1–20. http://arxiv.org/abs/1908.09168

[4] Alsaffar, D. M., Sultan Almutiri, A., Alqahtani, B., Alamri, R. M., Fahhad Alqahtani, H., Alqahtani, N. N., Mohammed alshammari, G., & Ali, A. A. (2020). Image Encryption Based on AES and RSA Algorithms. 2020 3rd International Conference on Computer Applications & Information Security, 1–5. https://doi.org/10.1109/IICCAIS48893.2020.9096809

[5] Arrag, S., Hamdoun, A., Tragha, A., & Khamlich Salah, E. (2013). Implementation of stronger AES by using dynamic S-box dependent of master key. Journal of Theoretical and Applied Information Technology, 53(2), 196–204.

[6] Bhoge, J. P., & Chatur, P. N. (2014). Avalanche Effect of AES Algorithm. International Journal of Computer Science and Information Technologies, 5(3), 3101–3103. http://ijcsit.com/docs/Volume5/vol5issue03/ijcsit2014050394.pdf

[7] Chennam, K. K., Muddana, L., & Aluvalu, R. K. (2017). Performance analysis of various encryption algorithms for usage in multistage encryption for securing data in cloud. RTEICT 2017 - 2nd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, Proceedings, 2018-Janua, 2030–2033. https://doi.org/10.1109/RTEICT.2017.8256955

[8] D’souza, F. J., & Panchal, D. (2017). Advanced encryption standard (AES) security enhancement using hybrid approach. Proceeding - IEEE International Conference on Computing, Communication and Automation, ICCAA 2017, 2017-Janua, 647–652. https://doi.org/10.1109/ICCAA.2017.8229881

[9] Daemen, J., & Rijmen, V. (2002). The Design of Rijndael. In New York. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-04722-4

[10] Das, I., Nath, S., Roy, S., & Mondal, S. (2012). Random S-Box generation in AES by changing irreducible polynomial. Proceedings of the 2012 International Conference on Communications, Devices and Intelligent Systems, CODIS 2012, 2, 556–559. https://doi.org/10.1109/CODIS.2012.6422263

[11] Dewangan, C. P., Agrawal, S., Mandal, A. K., & Tiwari, A. (2012). Study of avalanche effect in AES using binary codes. Proceedings of 2012 IEEE International Conference on Advanced Communication Control and Computing Technologies, ICACCT 2012, 978, 183–187. https://doi.org/10.1109/ICACCT.2012.6320767

[12] Elazm, L. A., El-Moursy, M. A., Elsimary, H., Dessouky, M. I., & Shawki, F. (2010). High speed low power composite field SBOX. IDT’10 - 2010 5th International Design and Test Workshop, Proceedings, 24–27. https://doi.org/10.1109/IDT.2010.5724400

[13] Ganesh, E. S., Velayutham, R., & Manimegalai, D. (2012). A secure software implementation of nonlinear AES S-box with the enhancement of biometrics. 2012 International Conference on Computing, Electronics and Electrical Technologies, ICCEET 2012, 927–932. https://doi.org/10.1109/ICCEET.2012.6203796

[14] Hosseinkhani, R. (2012). Using Cipher Key to Generate Dynamic S-Box in AES Cipher System. International Journal of Computer Science and Security (IJCSS), Volume (6) : Issue (1) : 2012, 6, 19–28.

[15] Hung, P. A., Klomkarn, K., & Sooraksa, P. (2013). Image encryption based on chaotic map and
dynamic S-box. *ISPACS 2013 - 2013 International Symposium on Intelligent Signal Processing and Communication Systems*, 435–439. https://doi.org/10.1109/ISPACS.2013.6704589

[16] Huy, D. Q., Duc, N. M., Khai, L. D., & Lung, V. D. (2019). Hardware implementation of AES with S-Box using composite-field for WLAN systems. *RVF 2019 - Proceedings: 2019 IEEE-RVF International Conference on Computing and Communication Technologies*, 1–6. https://doi.org/10.1109/RVF.2019.8713711

[17] Jithendra, K. B., & Shahana, T. K. (2019). A New Efficient Sbox for Strengthening PRESENT Like Block Ciphers Against Linear Cryptanalysis. *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies, ICICICT 2019*, 507–511. https://doi.org/10.1109/ICICICT46008.2019.8993397

[18] Juremi, J., Mahmood, R., & Sulaiman, S. (2012). A proposal for improving AES S-box with rotation and key-dependent. *Proceedings 2012 International Conference on Cyber Security, Cyber Warfare and Digital Forensic, CyberSec 2012*, 38–42. https://doi.org/10.1109/CyberSec.2012.6246172

[19] Kazlauskas, K., & Kazlauskas, J. (2009). Key-Dependent S-Box Generation in AES Block Cipher System. *Informatica*, 20(1), 23–34. https://doi.org/10.15388/informatica.2009.235

[20] Kumar, A., & Tiwari, N. (2012). Effective implementation and evaluation of AES in Matlab. *Advances in Intelligent Systems and Computing, 176 AISC(VOL. 1)*, 95–101. https://doi.org/10.1007/978-3-642-31513-8_11

[21] Lee, H., Lee, K., & Shin, Y. (2010). Implementation and performance analysis of AES-128 CBC algorithm in WSNs. *International Conference on Advanced Communication Technology, ICACT, 1*, 243–248.

[22] Li, L., & Li, S. (2017). High throughput AES encryption/decryption with efficient reordering and merging techniques. *2017 27th International Conference on Field Programmable Logic and Applications, FPL 2017*, 12–15. https://doi.org/10.23919/FPL.2017.8056803

[23] Manjula, G., & Mohan, H. S. (2018). Improved Dynamic S-Box generation using Hash function for AES and its Performance Analysis. *Proceedings of the 2nd International Conference on Green Computing and Internet of Things, ICGCIoT 2018, November 2001*, 109–115. https://doi.org/10.1109/ICGCIoT.2018.8753074

[24] Mohamed, K., Mohammed Pauzi, M. N., Hj Mohd Ali, F. H., Ariffin, S., & Nik Zulkipi, N. H. (2014). Study of S-box properties in block cipher. *I4CT 2014 - 1st International Conference on Computer, Communications, and Control Technology, Proceedings, I4ct*, 362–366. https://doi.org/10.1109/I4CT.2014.6914206

[25] Nejad, F. H., Sabah, S., & Jam, A. J. (2014). Analysis of avalanche effect on advance encryption standard by using dynamic S-Box depends on rounds keys. *2014 International Conference on Computational Science and Technology, ICCST 2014, 2014(2005)*. https://doi.org/10.1109/ICCST.2014.7045184

[26] Park, T., Seo, H., Kim, J., Park, H., Kim, H., & Kim, C. H. (2018). Efficient Parallel Implementation of Matrix Multiplication for Lattice-Based Cryptography on Modern ARM Processor. *Security and Communication Networks, 2018*. https://doi.org/10.1155/2018/7012056

[27] Patil, P., Narayankar, P., Narayan, D. G., & Meena, S. M. (2016). A Comprehensive Evaluation of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish. *Procedia Computer Science, 78*(December 2015), 617–624. https://doi.org/10.1016/j.procs.2016.02.108

[28] Schneier, B., & Sutherland, P. (1995). *Applied Cryptography: Protocols, Algorithms, and Source Code in C* (2nd ed.). John Wiley & Sons, Inc.

[29] Shannon, C. E. (1998). Communication theory of secrecy systems. 1945. *M.D. Computing: Computers in Medical Practice, 15*(1), 57–64.
[30] Shi, H., Deng, Y., & Yu, G. (2011). Analysis of the avalanche effect of the AES S box. *2011 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce, AIMSEC 2011 - Proceedings*, 5425–5428. https://doi.org/10.1109/AIMSEC.2011.6009935

[31] Shivkumar, S., & Umamaheswari, G. (2011). Performance Comparison of Advanced Encryption Standard (AES) and AES key dependent S-box - Simulation using MATLAB. *Proceedings of 2011 International Conference on Process Automation, Control and Computing, PACC 2011*, 1–6. https://doi.org/10.1109/PACC.2011.5979007

[32] Shreenivas Pai, N., Raghuram, S., Chennakrishna, M., & Karthik, A. S. V. (2017). Logic optimization of AES S-Box. *International Conference on Automatic Control and Dynamic Optimization Techniques, ICACDOT 2016*, 1042–1046. https://doi.org/10.1109/ICACDOT.2016.7877745

[33] Singh, A., Agarwal, P., & Chand, M. (2019). Image Encryption and Analysis using Dynamic AES. *2019 International Conference on Optimization and Applications, ICOA 2019*, 1–6. https://doi.org/10.1109/ICOA.2019.8727711

[34] Tran, B. N., Nguyen, T. D., & Tran, T. D. (2009). A New S-Box structure to increase complexity of algebraic expression for block cipher cryptosystems. *ICCTD 2009 - 2009 International Conference on Computer Technology and Development, 2*(2), 212–216. https://doi.org/10.1109/ICCTD.2009.235

[35] Ueno, R., Homma, N., Morioka, S., Miura, N., Matsuda, K., Nagata, M., Bhasin, S., Mathieu, Y., Graba, T., & Danger, J. L. (2020). High Throughput/Gate AES Hardware Architectures Based on Datapath Compression. *IEEE Transactions on Computers, 69*(4), 534–548. https://doi.org/10.1109/TC.2019.2957355

[36] Vaidehi, M., & Rabi, B. J. (2014). Design and analysis of AES-CBC mode for high security applications. 2nd International Conference on Current Trends in Engineering and Technology, ICCTET 2014, 499–50. *2nd International Conference on Current Trends in Engineering and Technology, ICCTET 2014*, 499–502. https://doi.org/10.1109/ICCTET.2014.6966347

[37] Webster, A. F., & Tavares, S. E. (1986). On the Design of S-Boxes. In H. C. Williams (Ed.), *Advances in Cryptology --- CRYPTO '85 Proceedings* (pp. 523–534). Springer Berlin Heidelberg.