JOURNAL OF SCIENCE TECHNOLOGY AND EDUCATION 12(2), JUNE, 2024
ISSN: 2277-0011; Journal homepage: www.atbuftejoste.com.ng

Dynamic Randomized Advanced Encryption Standard (DR-AES): A Solution to Enhance the Security of Mobile Learning System

[1]Adamu, M., [2]Oyefolahan, O. I., [3]Ojerinde, O. A., [4]Abdulmalik M. D.
[1]Department of Computer Science, Federal Polytechnic, P.M.B. 55, Bida, Niger State
[2]Department of Information Technology, Federal University of Technology, Minna, Niger State
[3]Department of Computer Science, Federal University of Technology, Minna, Niger State
[4]Department of Computer Science, Federal University of Technology, Minna, Niger State

ABSTRACT
The Advance Encryption Standard (AES) is a widely used encryption algorithm that provides strong protection for sensitive data. The AES is the most popular symmetric cryptographic scheme that uses the same key for encryption and decryption process. However, the conventional AES suffers from key distribution and resources consumptions problem. One of the key performance issues of the traditional AES algorithm is its complexity and speed of execution. These issues made AES unsuitable for real-time applications and applications that required less complexity. This paper proposes a new enhance AES encryption scheme called Dynamic Random Advance Encryption Standard (DR-AES) and implement in MATLAB environment. The DR-AES introduces randomness into the encryption process, making it more difficult for attackers to launch certain types of attacks and reduced number of rounds to reduce the complexity of AES and make it suitable for securing sensitive information of mobile learning system less. The performance of the developed DR-AES is evaluated in terms of encryption time, decryption time, and avalanche effect. The result suggest that the developed DR-AES improved the time required for encrypt and decrypt drastically reduced to minimal level and achieved a reasonable avalanche effect of higher percentage thereby increases the level of complexity of the transmission, making decryption of transmitted information more difficult and complicated to intruders. From a security point of view, the proposed DR-AES complies with Kerkhof's principle, and its performance proved better. This means the scheme has open design and the security provided by the DR-AES depends only on the secrecy of the encryption key.

ARTICLE INFO
Article History
Received: January, 2024
Received in revised form: March, 2024
Accepted: March, 2024
Published online: June, 2024

KEYWORDS
Advanced Encryption Standard, Cryptography, Dynamic Encryption, Cipher Text, Plain Text, Encryption, Decryption.

INTRODUCTION

Security threats plague mobile learning systems in educational institutions, despite their vast potential for motivating learners and facilitating accessibility and collaboration. These systems offer opportunities for distributed and innovative learning, yet concerns over security are escalating, particularly in regions like Nigeria, where cybercrime is rampant. The prevalence of security issues poses significant challenges to the effective operation of mobile learning systems in such environments (Mseteka & Phiri, 2019). Mobile technology, crucial for mobile learning, facilitates vast data transfer across industries, yet insecure channels like wireless networks pose vulnerability. To safeguard academic data, both public and private educational sectors employ diverse security techniques (Chinedu *et al*., 2020; Aliu *etal*., 2021). Cryptography stands out as a vital method, utilizing encryption to protect data integrity from potential attackers (Abdullah, 2017).

Encryption safeguards sensitive data by transforming it into unreadable ciphertext via

Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.

algorithms like byte substitutions and matrix transformations. The choice of encryption algorithm is vital for information security, with various commonly available options. However, the selection of an appropriate key is equally crucial as it directly influences the encryption's security level (D'souza, 2017). Symmetric key encryption algorithms like DES, Triple DES, and AES are widely utilized for data protection. DES, a block cipher developed in 1977, emerged from a US National Bureau of Standards competition. AES, comprising bit substitutions and round key functions, is a swift and resilient encryption method. It remains unbroken by practical attacks, making it a stalwart choice for advanced information security. (Nivedhaa & Justus, 2018).

The AES algorithm has multiple options for configuration. The first option is the key size or key length, which offers three choices: 128, 192, and 256 bits. The second option is the chaining mode, which determines how multiple block ciphers operate in a chained mode to encrypt information larger than 16 bytes. Some common chaining modes include: Electronic Code Book (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), and Output Feedback (Soumya *et al.,* 2018; Momoh *et al.,* 2021).

Several modifications have been carried out on AES algorithm, most of which were done to improve the AES implementation in terms of memory reduction, computational power minimization, encryption and decryption time reduction and security. However, most of this research still uses maximum AES rounds which is still complex for light weight or resource starved applications. Hence, this paper proposes a new Dynamic Randomized AES to reduce the number of rounds required for encryption and decryption while improving the security for mobile devices. The remainder of this paper is structured as follows: Section 2 discuss the existing modifications of AES and the traditional AES, section 3 discussed the proposed DR-AES scheme. In section 4, the experimental results are discussed. This is followed by conclusion in section 5.

Literature Review

Several modifications to enhance AES encryption scheme has been proposed. Abikoye *et al.,* (2019) proposed an enhanced version of AES by modifying the SubBytes and ShiftRows transformations. The SubBytes transformation was made round key dependent, while the ShiftRows transformation was randomized. The goal of these modifications was to make the two transformations dependent on the key, so that a small change in the key would result in a significant change in the ciphertext. Altigani *et al.,* (2018) introduced a dynamic algorithm that defines the exact steps used to encrypt or decrypt payloads at runtime. This was achieved by entering AES parameters instead of reusing fixed and default values. The resulting encryption works very similar to AES. However, extensive implementation of this technique is required to properly analyze the performance costs of these changes.

Sikarwar & Murty (2018) Presents an AES implementation using a low-power shift register with an ADOC and RTPG triggering technique that improves the performance of the implemented AES design. The reason for choosing a shift register architecture when multiple architectures are available is the low power consumption and small footprint compared to other architectures. Talirongan *et al*. (2019) introduced a novelty regarding the modification of the circular function of the AES algorithm to increase the security of encryption and decryption processes using the butterfly effect. The butterfly effect indicates that small causes can have larger effects. Several studies have demonstrated the use of the butterfly effect used to calculate temperature dependence using holographic matching. Also in the process, "Add Butterfly Effect" is added on each round. These results suggest that the modified AES has improved the level of complexity of the transmission, making decryption of user information more difficult and complicated. Although, the butterfly effect may experience lack of sensitivity to small changes in the encryption process.

Altigani *et al*. (2021) introduced a polymorphic variant of the Advanced Encryption Standard. With P-AES, the values of the AES parameters change with each new key. Exact values are only available to authorized communication partners at runtime. To achieve these goals, the fundamental transformations of AES, SubBytes, ShiftRows and Mix Columns in

Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.

the proposed P-AES were crucial. The proposed P-AES can support 16, 24- or 32-byte keys. However, for consistency, the key length is assumed to be 32 bytes. (Assafli & Hashim, 2020) propose enhancing AES security in CBC mode with a new technique, leveraging Unix time as the IV source. Through experimentation, they demonstrate that this method meets strict criteria 53% of the time, bolstering ciphertext strength compared to traditional AES-CBC. Despite its efficacy, the encryption's error propagation property poses a risk of sender-receiver synchronization loss. The assessment using the snowball effect validates the performance of the improved AESCBC technique, emphasizing its potential in enhancing data security.

Zinabu and Asferaw (2022) introduce an enhancement to AES called EE-AES, substituting the mix column stage with a bitwise reverse transposition method. This modification reduces computational demands while preserving AES's security level. EE-AES employs hexadecimal notation to map two hex digits to eight binary bits, enhancing speed and efficiency compared to AES and MAES. The proposed algorithm's potential could be further explored by testing it with various bit sizes and comparing it with other modern encryption methods. This innovation highlights a promising avenue for improving encryption efficiency without compromising security.

Dynamic Randomised Advanced Encryption standard (DR-AES)

Based on the limitations of the existing conventional AES and enhanced AES ciphers namely: intolerance performance degradation, explicit conflict with Kerckhoff's principle and inter-operability challenges, questionable security challenges, and high implementation cost and these issues made the available AES and modified AES unsuitable for real-time applications and applications that required less complexity such as lightweight devices (i.e. mobile devices).

To address these issues, the research study aims at developing a novel Dynamic Randomized AES (DR-AES) encryption scheme that can provide practical solutions to the fore mentioned limitations. The modification of AES is based on reducing the number of rounds and dynamically randomized the add-round key operations. By reducing the number of rounds, the speed of the algorithm will greatly improve and make it suitable for light weight and real-time applications such as mobile learning information security. To compensate on the effect of reducing the number of rounds on the security, a randomized add round key is introduced.

In this process, the initial secrete key is expanded using the standard key expansion algorithm. The 128-bits key is expanded to produce 11 new 128 bits (16-bytes) which corresponds with the number of rounds plus one for the preliminary round. For each of the number of chosen rounds (nR), the appropriate round key corresponding to the round number will be selected for the add round operation to change the state. Each 16 byte of the expanded key will be indexed for reference. Figure 1 shows the proposed DR-AES scheme for encryption and decryption.

Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.

JOURNAL OF SCIENCE TECHNOLOGY AND EDUCATION 12(2), JUNE, 2024
ISSN: 2277-0011; Journal homepage: www.atbuftejoste.com.ng

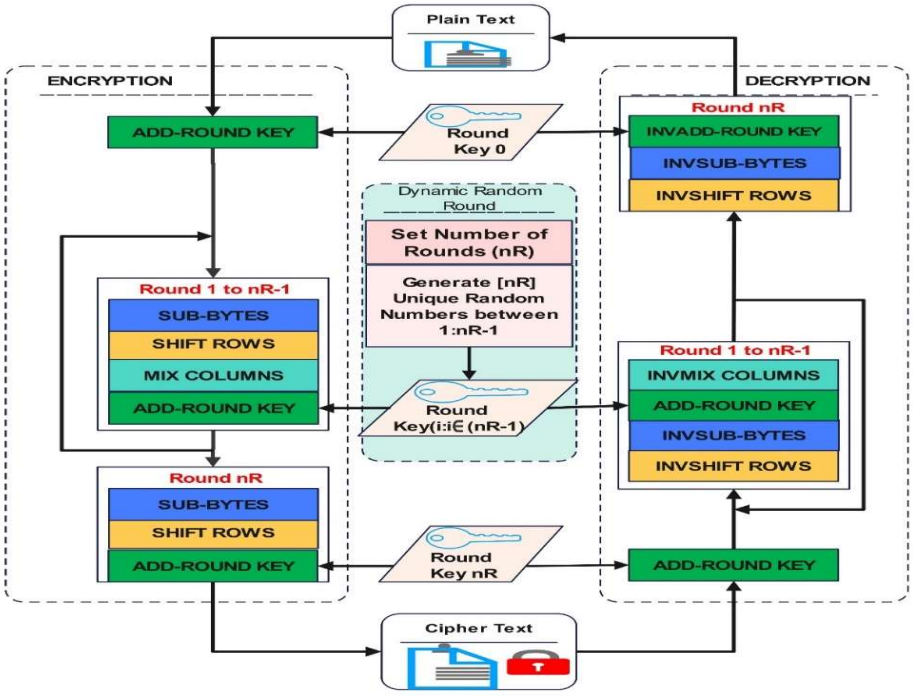


Figure 1: Randomized Dynamic AES Scheme

The proposed scheme comprises two basic steps: the encryption and decryption steps. The encryption steps are made up of 4 basic steps: Key Generation and Expansion, Dynamic Random Round Generation, Add-Round Key, and Round Function (Sub-bytes, Shift Rows, Mix Columns, and Add-Round Key).

Key Generation and Expansion

The key expansion used in this research work was adopted from (Yan & Chen, 2016) with some modification. The initial 128-bit key is divided into 4 words and then further expanded into 44 words. Each word is arranged into a 4x4 matrix, with 4 words representing a single key. A total of 11 subkeys are needed, and each key is expressed in 16 bytes. The first key is represented as W_0, W_1, W_2, and W_3. An index number r_0 to r_{10} was assigned to each word as shown in Figure 2.

Figure 2: Key expansion

Add Round Key

Add round key: Add Round Key transformation is an XOR operation that adds the round key to the State in each round. In this step, the round key, which is derived from the original encryption key, is added to the state using a bitwise XOR operation. This step is used to introduce confusion into the encryption process. This step is important because it is a non-linear operation that mixes the plaintext with the key, making it more difficult for an

Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.

attacker to decrypt the cipher text without the correct key.

Random Round Generation

The dynamic random round generation stage comprises of three steps:

i. Determination of the number of rounds (nR)
ii. Generation of nR unique random numbers between 1: N. where N is the total number of rounds corresponding to the key length. In this work, 128-bit key is proposed given rise to 10 rounds (N=10).
iii. Select appropriate round key for add round function.

.

Round Function

The round function step is made up of 4 basic steps which are sub-bytes, shift rows, mix columns and add round keys.

Sub-bytes

The Sub-bytes transformation is a nonlinear byte substitution that operates on each byte of the State using a table, known as an S-box. The S-box is computed using a finite field inversion and an affine transformation. In this step, each byte in the state is replaced with a corresponding value from the S-box. Figure 3 shows the implementation of the sub-byte operations that is applied to generate the state 3 matrix. The input is the initial state array from the initial add-round operation

Figure 3: Sub-byte

Shift rows

The Shift Rows transformation is a circular shifting operation that rotates the rows of the State with different numbers of bytes (offsets). In this step, the rows of the state are shifted by a certain number of positions. The number of positions shifted depends on the row number, with the first row being un-shifted, the second row being shifted one position, and the third row being shifted two positions, and so on. Figure 4 shows the sub-byte process.

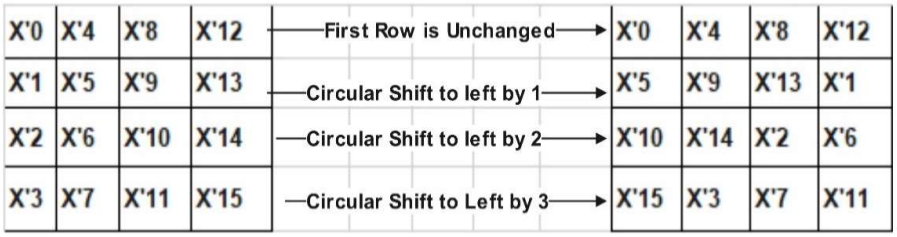


Figure 4: Shift rows process.

Mix Columns

In mix columns, each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round. This step is used to mix the columns of

Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.

the transformed data before it is encrypted. Figure 5 depicts the transformation operation of the mix column function.

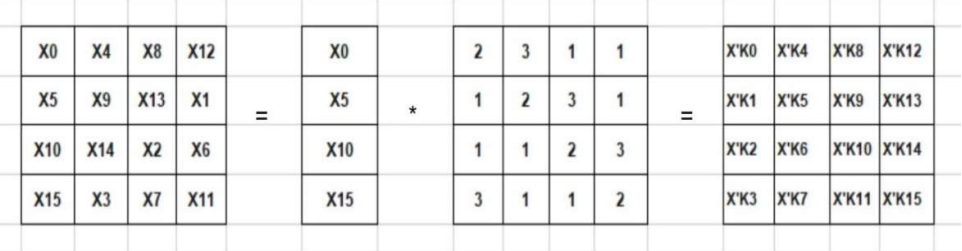


Figure 5: Mix Columns

Add round key

Add Round Key transformation is an XOR operation that adds the round key to the State in each round. In this step, the round key, which is derived from the original encryption key, is added to the state using a bitwise XOR operation. This step is used to introduce confusion into the encryption process. This step is important because it is a non-linear operation that mixes the plaintext with the key, making it more difficult for an attacker to decrypt the cipher text without the correct key. Figure 6 shows the add round key transformation process.

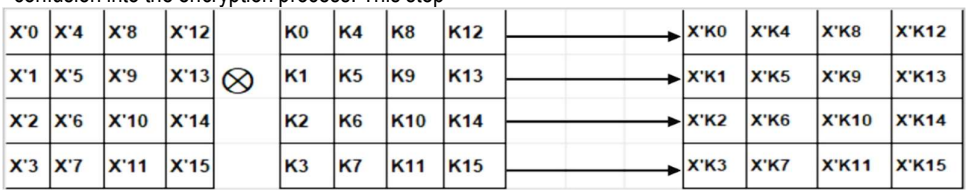


Figure 6: Add round key

(DR-AES) Encryption Algorithm

Table 1 shows the pseudocodes for encryption algorithm which defined the steps of the proposed DR-AES encryption process operation. Plaintext and key are the input while, cipher text is the output of the encryption transformation.

Table 1: Encryption Algorithm

ALGORITHM: DR-AES ENCRYPTION ALGORITHM
INPUT: PLAINTEXT, KEY
OUTPUT: CIPHER TEXT

1. *Start*
2. *Define plaintext*
3. *Define key*
4. *Call the function tic to start a timer to measure the execution time.*
5. *Convert key to hexadecimal and assign the result to variable KEY.*
6. *Convert plaintext to hexadecimal and assign the result to variable PLAINTEXT.*
7. *Calculate the number of words in the key by dividing the length of KEY by 8 and assigning the result to variable Nk.*
8. *Call the function KeyExpansion with arguments KEY and Nk to expand the key and assign the result to variable w.*
   a. *Convert the key from hexadecimal to decimal using the hex2dec function and reshape it into a 2D array with 2 columns.*

Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.

b. Reshape the key from step a into a 4xNk matrix.
c. For i from Nk to 4(Nk+7)-1 do the following:*
i. Set the temporary variable temp to the ith column of w.
ii. If i is a multiple of Nk then:
1. Shift the ith column of w circularly to the right by 1 and apply the SubBytes function to it.
2. Calculate the polynomial n = xtime(2, n) where n is the last byte of temp and xtime is a function that multiplies its input by 2 in the Galois field GF(2^8). 3. Bitwise XOR the last byte of temp with n and set the result to the last byte of temp.
iii. Else if Nk > 6 and i is a multiple of 8 then apply the SubBytes function to temp.
iv. Bitwise XOR the ith column of w with the (i-Nk)th column of w and temp and set the result to the (i+1)th column of w.
d. Return w.

9. *Randomly select 3 values from the range 1 to 10 and assign them to variable DR.*
10. *Call the function tic to start a timer to measure the execution time.*
11. *Call the function newCipher with arguments PLAINTEXT, R, and w to encrypt the plaintext.*
    a. *Convert In from hexadecimal to decimal and reshape the result to a matrix with 4 rows and a number of columns.*
    b. *Assign the result to variable state.*
    c. *Call the function AddRoundKey with arguments state and the first 4 columns of w to add the key to the state.*
        1. *Define the S-box matrix using hexadecimal values*
        2. *Reshape the S-box matrix by transposing it and then concatenating the rows*
        3. *Convert the concatenated matrix from hexadecimal to decimal using hex2dec function*
        4. *Reshape the decimal matrix into a 16x16 matrix*
        5. *Define the input state matrix*
        6. *Substitute each element of the state matrix with the corresponding value from the S-box matrix using the formula state=Sbox(state+1)*
        7. *Perform row shifts on the state matrix using the function ShiftRows(state)*
        8. *Perform column mixing on the state matrix using the function MixColumns(state)*
    d. *Loop from k=1 to NR-1, where NR is the number of values in R.*
    e. *Call the function SubBytes with argument state to substitute bytes in the state.*
    f. *Call the function ShiftRows with argument state to shift the rows in the state.*
    g. *Call the function MixColumns with argument state to mix the columns in the state.*
    h. *Call the function AddRoundKey with arguments state and the columns of w specified by DR(k) and DR(k)+1 to add the round key to the state.*
    i. *End loop.*
    j. *Call the function SubBytes with argument state to substitute bytes in the state.*
    k. *Call the function ShiftRows with argument state to shift the rows in the state.*
    l. *Call the function AddRoundKey with arguments state and the columns of w specified by R(end) and R(end)+1 to add the round key to the state.*
    m. *Reshape state to a column vector and assign the result to variable Out1.*
    n. *Convert Out1 to lowercase hexadecimal.*
    o. *Convert Out1 from a column vector to a row vector.*
12. *Assign the result to variable Out1 (**CIPHERTEXT).***
13. *Call the function toc to stop the timer and calculate the encryption time. Assign the result to variable encrypt Time.*
14. *End*

Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.

(DR-AES) Decryption Algorithm

Table 2 shows the pseudocodes for decryption algorithm which spelled the steps of the proposed DR-AES decryption process operation. The cipher text and dynamic round (DR) along with a key expansion (w) are the input while, plain text is the output of the decryption transformation.

Table 2: Decryption algorithm

ALGORITHM: DR-AES DECRYPTION ALGORITHM
INPUT: CIPHERTEXT, DR, w
OUTPUT: PLAIN TEXT

1. *Start*
2. *Define a function named **newInvCipher** that takes in two inputs, **CIPHERTEXT** and **DR**, along with a key expansion **w**.*
3. *Determine the size of the **DR** matrix and store the value in the variable **NR**.*
4. *Convert the input **CIPHERTEXT** from a hexadecimal string to a decimal array using the **hex2dec** function and reshape it into a 4x4 matrix. Store the result in the variable **state**.*
5. *Perform an initial **AddRoundKey** operation by selecting the key schedule **w** corresponding to the last round (**DR (end)**) and XOR it with the **state**.*
6. *Iterate through the remaining rounds in reverse order from **NR-1** down to **1**.*
7. *Perform an **InvShiftRows** operation on the **state**.*
8. *Perform an **InvSubBytes** operation on the **state** using the inverse S-Box lookup table defined in the **Sbox** function.*
9. *Perform an **InvMixColumns** operation on the **state**.*
10. *Repeat steps 6-9 for all rounds except the first.*
11. *Convert the **state** matrix back into a 1x16 decimal array and convert it to a lowercase hexadecimal string using the **dec2hex** and **lower** functions. Store the result in the variable **PLAIN TEXT**.*
12. *Return the **PLAIN TEXT** string as the output of the **newInvCipher** function.*
13. *Call the function toc to stop the timer and calculate the decryption time. Assign the result to variable decrypt Time.*
14. *End*

EXPERIMENTAL RESULT AND DISCUSSION

``The developed algorithms of DR-AES scheme have been implemented using MATLAB tool.

Encryption and Decryption Time of DR-AES

The encryption and decryption performance are a major metric that measure the success or failure of any cipher. Even the strongest cipher will be deemed impractical if it has poor performance.

Average Encryption and Decryption Time of DR-AES

The average times provide an overall summary of the performance across all configurations. The table 3 consist of the results of the average time of encryption and decryption process for each of number of round selections of the developed DR-AES scheme. The result in the table reveals the least the number of rounds the fastest the time it takes the cipher to encrypt and decrypt and this is in line with the principle of cryptography. Conversely, the more the number of rounds the more the time (slow) time it times it takes the cipher to encrypt and decrypt.

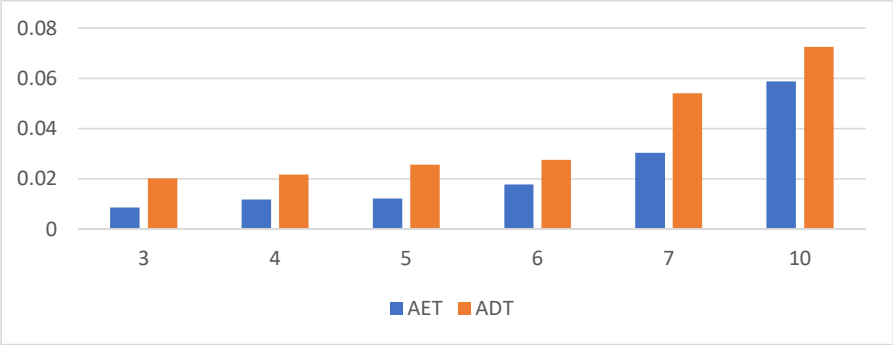
Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.

Table 3: Average Encryption and Decryption Time of DR-AES

Number of Round (NR)	Average Encryption Time	Average Decryption Time
3	0.0087	0.0201
4	0.0118	0.0217
5	0.0122	0.0257
6	0.0178	0.0276
7	0.0305	0.0540
10	0.0588	0.0726

Graphical Presentation of Average Time for DR-AES Encryption and Decryption Process

The figure 7 is the graphical presentation of the encryption and decryption time of the DR-AES. The graph reveals the least the number of rounds the fastest the time it takes the cipher to encrypt and decrypt and this is in line with the principle of cryptography. Conversely, the more the number of rounds the more the time (slow) time it times it takes the cipher to encrypt and decrypt. The three (NR=3) rounds have the least average encryption and decryption time, the next to NR=3 round is NR=4 followed by NR=6 and NR=7 is the next after NR6. While, NR=10 rounds have the highest average encryption and decryption time. The delay in ten rounds is attributed to the higher number of rounds involves in the process of execution which may be suitable for quick respond or real time applications. The least time taken by three (NR=3) rounds makes 3DR-AES the fastest among others number of rounds. Hence, the NR=3 rounds scheme is the most preferable choice for mobile devices with limited resources.



Avalanche Effect Result

Avalanche effect is one the most power factor used in measuring the strength of any new cipher in terms of security resistance to attacks. It measures how much the cipher output will change if a small change is introduced to the input. It is highly expected in any cipher to have avalanche score not less than 50% (Altigani et al., 2021). This means, if only one bit is change in the cipher input, then every bit in the cipher output has a probability of not less than 50% to change it value.

Overall Avalanche Effect Performance Comparison of DR-AES

Table 4 captured the avalanche effect result for all the dynamic number of rounds of the DR-AES for the test for 1 bit, 2 bits and 3 bits change in the plaintext. The revealed that the avalanche effect score increases as the number of rounds increases for each case of number of bit tempered in the plain text. The same cipher highly unpredictable in terms of key secrecy as can be seen the greater number of bit change in the plain text, the higher significant change in the cipher text. The result is in lined with Mathur et al., (2020) that highlighted that the security strength of the cryptography

Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.

protocol depends on the key size, which implies, the larger the key size, the very difficult to corrupt the key.

Table 4: Overall Avalanche Effect of DR-AES

Number of Round (NR)	1-bit Change	2-bits Change	3-bits Change
3	53.32%	54.07%	55.13%
4	54.18%	55.13%	56.72%
5	55.32%	56.47%	57.72%
6	56.60%	57.39%	58.39%
7	57.58%	58.34%	59.31%
10	58.33%	59.06%	60.20%

Overall Avalanche Effect Comparison Chart of DR-AES

Figure 8 depicts the results of the various number of rounds of the developed DR-AES. The figure shows the higher the number of rounds, the higher the avalanche scores and this is cut across all the number of bit change in the plain text experimented. The result is in lined with the research work of Mathur et al., (2020) that highlighted that the security strength of the cryptography protocol depends on the length of the key size, which implies, the larger the key size, the very difficult to corrupt the key.

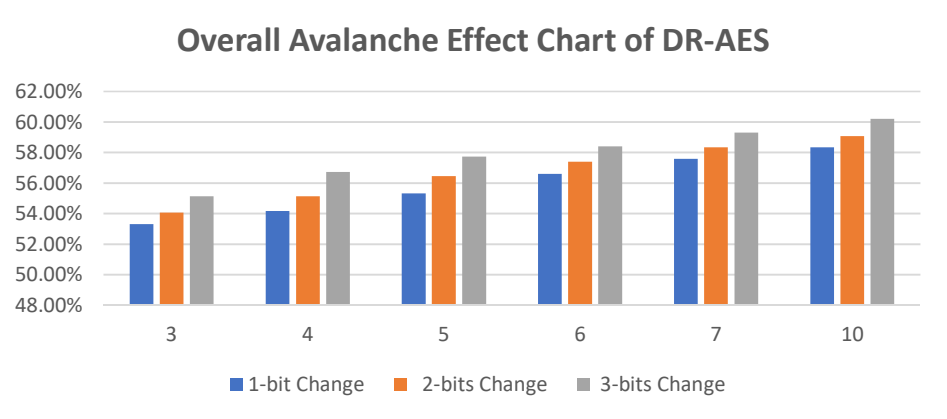


Figure 8: Overall Avalanche Effect Comparison Chart of DR-AES

CONCLUSION

The paper developed a dynamic randomized Advanced Encryption Standard, namely DR-AES. The study aims to changes the static features of the traditional AES and replaced it with dynamic randomized cipher that inherits the AES strengths in order to enhance the security of mobile learning system. The performance of the developed DR-AES is evaluated in terms of encryption time, decryption time, and avalanche effect. The result suggest that the developed DR-AES has improved as the time required for encrypt and decrypt drastically reduced to minimal level and also achieved a reasonable avalanche effect of higher percentage thereby increases the level of complexity of the transmission, making decryption of transmitted information more difficult and complicated to intruders. From a security point of view, the proposed DR-AES complies with Kerkhof's principle, and its performance proved better. This means the

Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.

scheme has open design and the security provided by the DR-AES depends only on the secrecy of the encryption key. In future work, throughput performance metric can be experimented to investigate the security performance without increasing the implementation cost and the need to deploy the developed DR-AES model into mobile learning environment.

REFERENCE

Abdullah, A. M. (2017). *Advanced Encryption Standard (AES) Algorithm to Encrypt and Decrypt Data*.

Abikoye, O. C., Haruna, A. D., Abubakar, A., Akande, N. O., & Asani, E. O. (2019). *SS symmetry for Information Security*. 1–16.

Assafli, H. T., & Hashim, I. A. (2020). Security enhancement of AES-CBC and its performance evaluation using the avalanche effect. *2020 3rd International Conference on Engineering Technology and Its Applications, IICETA 2020*, 7–11. https://doi.org/10.1109/IICETA50496.2020.9318803

Aliu, D., Shaba M. S., Momoh, M. O., Chinedu, P U., & Nwankwo, W A computer security system for cloud computing based on encryption technique. *Computer Engineering and Applications Journal*, *10*(1), 41-54. A computer security system for cloud computing based on encryption technique. Computer Engineering and Applications Journal, 10(1), 41-54.

Altigani, A., Hasan, S., Barry, B., Naserelden, S., Elsadig, M. A., & Elshoush, H. T. (2021). A Polymorphic Advanced Encryption Standard - A Novel Approach. IEEE Access, 9, 20191–20207. https://doi.org/10.1109/ACCESS.2021.3051556

Assafli, H. T., & Hashim, I. A. (2020). Security enhancement of AES-CBC and its performance evaluation using the avalanche effect. 2020 3rd International Conference on Engineering Technology and Its Applications, IICETA 2020, 7–11. https://doi.org/10.1109/IICETA50496.2020.9318803

Chinedu, P. U., Nwankwo, W., Aliu, D., Shaba, S. M., & Momoh, M. O. (2020). Cloud security concerns: assessing the fears of service adoption. Archive of Science and Technology, 1(2), 164-174.

D'souza, F. J., & Panchal, D. (2017). Advanced Encryption Standard (AES) Security Enhancement using Hybrid Approach. 647–652. García,D.F. (2015).Standard Algorithm. 247–252. https://doi.org/10.1109/MCSI.2015.61

Momoh, M. O., Chinedu, P. U., Nwankwo, W., Aliu, D., & Shaba, M. S. (2021). Blockchain Adoption: Applications and Challenges. *International Journal of Software Engineering and Computer Systems*, *7*(2), 19-25.

Mseteka, L., & Phiri, J. (2019). A Secure Model for Storage and Dissemination of Examination Results: A Case Study of Zambia Technical Education Vocational and Entrepreneurship Training Authority. *Journal of Computer Science, 15*(2), 221–234.

Muttaqin, K., Rahmadoni, J., Samudra, U., & Andalas, U. (2020). Analysis And Design of File Security System AES (Advanced Encryption Standard) Cryptography Based. 1(2), 114–123.

Nivedhaa, R., & Justus, J. J. (2018). A Secure Erasure Cloud Storage system using Advanced Encryption Standard algorithm and Proxy Re-encryption. 2018 International Conference on Communication

Sikarwar, Y. S., & Murty, N. S. (2018). Low Power Implementation of Advanced Encryption Standard using Efficient Shift Registers in 45 nm Technology. *Proceedings of the 3rd International Conference on Communication and Electronics Systems, ICCES 2018, 2018-Janua*(Icces), 26–30. https://doi.org/10.1109/CESYS.2018.8723879

Soumya, V. H., Neelagar, M. B., & Kumaraswamy, K. V. (2018). Designing of AES Algorithm using Verilog. *2018 4th International Conference for*

Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.

Convergence in Technology, I2CT 2018, 1–5. https://doi.org/10.1109/I2CT42659.2018.9058322

Talirongan, H., Sison, A. M., & Medina, R. P. (2019). Modified advanced encryption standard using butterfly effect. 2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management, HNICEM 2018. https://doi.org/10.1109/HNICEM.2018.8666368

Yan, J., & Chen, F. (n.d.). An Improved AES Key Expansion Algorithm. *Proceedings of the 2016 International Conference on Electrical, Mechanical and Industrial Engineering*, 113–116. https://doi.org/https://doi.org/10.2991/icemie-16.2016.28

Zinabu, N. G., & Asferaw, S. (2022). Enhanced Efficiency of Advanced Encryption Standard (E E -AES) Algorithm. 7(mix), 59–65. https://doi.org/10.11648/j.ajetm.20220703.13

Corresponding author: Adamu, M.
bejian2004@gmail.com
Department of Computer Science, Federal Polytechnic, Bida, Niger State.