IRAQI Academic Scientific Journals

Samarra Journal of Pure and Applied Science

SJPAS

p ISSN: 2663-7405
e ISSN: 2789-6838

www.sjpas.com

Enhancing AES Security Based on Camellia Key Schedule

Arshad Sami Sulaiman*, Maytham M. Hammood
Department of Computer Science, College of Computer Science and Mathematics, University of Tikrit, Iraq

https://doi.org/10.54153/sjpas.2025.v7i1.1020

Article Information
Received: 21/08/2024
Revised: 15/09/2024
Accepted: 20/10/2024
Published: 30/03/2025

Keywords:
AES, Camellia, Key Schedule, NIST, Randomness.

Corresponding Author

E-mail:
arshad.s.sulaiman@st.tu.edu.iq
Mobile:

Abstract

The primary function of AES is security; this is where the Advanced Encryption Standard (AES) comes into play. Combining AES with an application significantly impacts its capacity to safeguard sensitive data. These include secure communications or private financial transactions. The adoption of AES in 2001 led to extensive evaluations that consistently demonstrated the effectiveness of AES in preventing attacks from all angles. The investigation suggests making the schedule of AES's keys more flexible to increase its resistance to direct attacks and other potential dangers. Another thing that the probe could consider is incorporating some of the information from Camellia's primary schedule of attacks with the AES algorithm; this would lead to a renaissance of Camellia's vulnerability to cyberattacks due to its complex design. The merge of these advanced Camellia Block Cipher properties onto the already standing AES structure could then bear fruit rich and plentiful. Later, we used the NIST-800-22 test suite to assess the randomness of the altered algorithm, and then we compared the new algorithm to the original AES. The outcomes show significant enhancements to the enhanced algorithm compared to the standard AES.

Introduction:

The Advanced Encryption Standard (AES) is a widely used encryption algorithm to secure sensitive data in many applications due to its strength of security and efficiency. It was established by the National Institute of Standards and Technology (NIST) in 2001. Since then, it has been examined in order to assess its effectiveness and resistance to various cryptographic attacks[1]. One unique aspect of AES is its secret key schedule, which is responsible for generating random keys for the rounds of encryption and decryption processes. Each round should have a unique random key. The random keys generated by the key schedule directly impact the randomness of encrypted data. So, any inefficiencies or flaws in the primary schedule can adversely affect the safety of any data encoded with these keys[2, 3]. Continuous research focuses on improving the AES key schedule to strengthen its resistance against various attacks, including related key attacks and differential attacks, by enhancing AES confusion properties, making it difficult for attackers to

threaten it[4]. Continuous advancements in the AES key schedule are required to ensure that it is still immune to continuously evolving threats. The Camellia is another block cipher (like AES) that is celebrated for its intricate yet long-lasting key scheduling, effectively avoiding consideration. The primary scheme of Camellia has been recorded in a standardized manner. ISO/IEC, endorsed by NESSIE and CRYPTREC, has been modified to increase safety while still providing effectiveness. The cipher is considered a versatile method that is successful in multiple scenarios. Both software and hardware implementations can meet the current requirements for cryptography[5-7]. While it has increased the difficulty of creating an AES schedule, this has also led to a more difficult process. Now, cryptographic methods are practical without negatively impacting performance. This is caused by the increased power of modern systems, which allows for a trade-off between data safety, efficiency, and effective performance while utilizing current techniques[8-11]. We chose the Camellia key schedule because Camellia has properties similar to AES in terms of block size, which is both 128 bits, and the available key sizes of 128, 192, and 256 bits. In addition, the Camellia algorithm has excellent performance compared to AES[12], and therefore, using its key schedule will be suitable for AES without a significant impact on performance. Previous investigations have attempted to alter the key schedule for the AES in order to increase security. All different designs created by Daemen and Rijmen are collectively referred to as The Wide Trail[13]. In this paper, we introduce an approach to enhancing the AES cryptographic algorithm with Camellia's strong key scheduling method; we chose Camellia because it has some of our research intends not just to bring another publication into the already manifold world of cryptography but to analyze the integration we propose in depth. We will examine two primary factors: firstly, the level of strength achieved by our cryptography; secondly, the impact it has on performance and the practical feasibility of implementing such a system. Expect that other researchers who are interested in improving encryption techniques and data protection in the future will utilize these findings as benchmarks.

The manuscript is organized into five sections. The first part talks about the most recent literature; the second presents the algorithms used; the third outlines the methodology we used in our work; the fourth shows our results; and finally, a conclusion is made in the fifth section.

Literature Review:

The Advanced Encryption Standard (AES) is commonly used in the field of information security because it has shown strong resistance against many threats and attacks. As the methods of attacking security systems keep advancing, it becomes very important to keep updating strategies that are used to safeguard these systems. Since AES plays a major role in protecting systems, its safety should be enhanced in every way possible. One effective way of doing this is by enhancing the key schedule, which forms a core part of the algorithm. This article presents some recent research done trying to come up with modifications that can be made to the AES Key Schedule, with the aim of increasing its resistance to attacks specific to this component of the algorithm.

A study introduces a novel approach to implementing AES-128 using three different keys. The updated AES showed increased encryption and decryption times, but the advantage gained is increased security, making data hacking require drastic efforts[14]. In another study, a new proposal was presented to improve the confidentiality of AES. It proposed a new method for key generation by using four functions: Gost external structure, Shift <<<, AES-Key Schedule, and MD5; the proposed approach overcomes the weaknesses in the original AES key schedule with enhanced confusion and diffusion properties in addition to increased randomness[15]. Another study introduces an advanced version of the AES algorithm with enhanced randomness. The proposed method used a modern version of the Fischer-Yates shuffle, which is a random number generator; this algorithm generates 1408-bit (11 rounds x 128-bit) keys. MATLAB software implementation is

used to evaluate the performance of this enhanced AES, and results are compared with those obtained using plain AES; the findings indicate improved security and performance [16]. In addition, this paper introduces improvements to AES key expansion using three ideas: the irreversible improvement technique, random number strategy, and word shift strategy. The results showed that the proposed method has better security and performance[17]. Another study explores the primary procedure for expanding the AES algorithm and makes specific alterations that enhance safety. The investigation demonstrates the necessity of secure scheduling of keys and raises concerns about the current method of expanding keys, which potentially lacks sufficient rigorous conditions. The model involves a mixture of affine recurrence and AES S-Box in the process of scheduling keys; this combination increases confusion[18]. Finally, a research paper suggests two modifications to AES to improve its security. The first modification is to generate the AES round keys relying on (DES key expansion, Substitutions layer, and Hash function MD5) where the Substitution layer adds confusion and MD5 adds diffusion properties; the second modification is to generate variable values to transfer as a matrix instead of the original key. The proposed algorithm, which used five statistical tests, showed enhancements in security over the original AES[19].

The most recent academic study examined various methods to enhance Advanced Encryption Standard (AES) security. The study focused on improving the procedure for scheduling keys. The review article emphasized improving the process of generating keys in round numbers in AES, which has been recognized as a significant area for future research in the field of cryptographic security.

Advanced Encryption Standard:

The core of symmetric key cryptography is a single key that manages encryption and decryption operations for all communication channels. The AES algorithm is a block cipher based on the SPN structure. It operates on data in fixed 128-bit blocks. The length of the key in bits (128, 192, 256) determines the level of security. The security is achieved through what is referred to as rounds—and the number of rounds is directly proportional to the size of the key: with a 128-bit key, 10 rounds are needed; with a 192-bit key, you will need 12 rounds, while a 256-bit key demands 14 rounds. These are essential prerequisites for both encryption and decryption processes[20]. Every iteration in the encryption process involves many changes to enhance the security of the data. The initial transformation consists of the substitution process, wherein each data byte is substituted with another byte derived from an S-box. This substitution serves to generate a state of confusion. The shift rows operation rearranges the components within rows to distribute one byte across several columns. On the other hand, the mix columns operation works on the columns of the state. It treats each column as a four-term polynomial and applies a fixed matrix (together with the column) to alter this polynomial. The concluding stage of each round involves the add key transformation. This process entails the amalgamation of a round-specific key, which is derived from the original key using a special key scheduling method. The objective of this technique is to generate distinct keys for each encryption iteration[21, 22]; as depicted in **Fig. 1**, AES is utilized to demonstrate the encryption process using a 128-bit key during decryption. The procedure utilizes the inverse version of each change to enable the data to be reverted back to its original state.

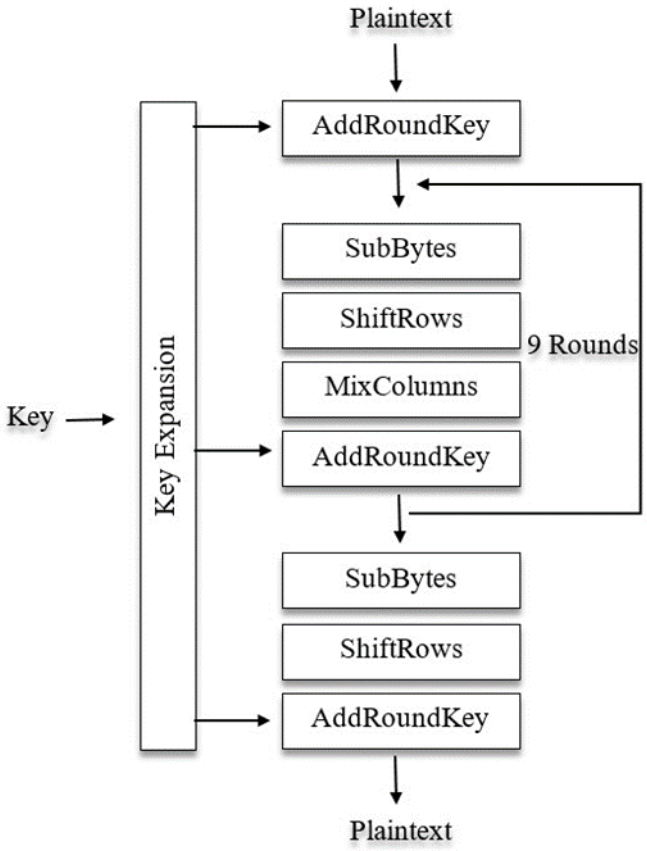

Fig. 1 AES encryption procedure for 128-bit

Camellia:

Camellia is a symmetric block cipher similar to AES in the sense that it operates on data blocks of 128 bits using the same secret key for both encryption and decryption operations. The cryptographic algorithm was jointly developed by Mitsubishi Electric and Nippon Telegraph and Telephone Corporation (NTT) in the early 2000s to find a balance between security and speed. This makes it appropriate for various purposes like safe communication and authentication. Despite Camellia's base on a Feistel-style network design, it augmented the effectiveness of encryption by combining the SPN method with a linear transformation approach. The volume of each block is consistently 128 bits; this does not alter. However, the length of the key varies and can be either 128, 192, or even 256 bits— this makes it quite versatile. Yet another feature that strengthens the algorithm's durability is its circular nature. The process of encrypting a 128-bit key involves 18 cycles. However, utilizing 192 or 256-bit keys increases the number of cycles to 24, increasing the key's potency for various length keys[8]. The Camellia encryption method is based on the F function as the primary component for encryption and decryption. Four different substitution boxes (S-boxes) are used in this method. These boxes support the phase and involve a rotation and XOR operation that works with Camellia's key scheduling like no other. To make the encryption harder and more secure, the algorithm employs FL and FL-1 functions every 6 rounds with specific parameters that include rotations, keyed XOR operations, and OR operations by certain bits' positions identified using those functions during each round[23], the effectiveness of Camellia in safeguarding digital data is demonstrated through different cryptographic techniques shown in **Fig. 2**, with a 128-bit key size example procedure depicted. There are many ways to do this, but only if you know how.

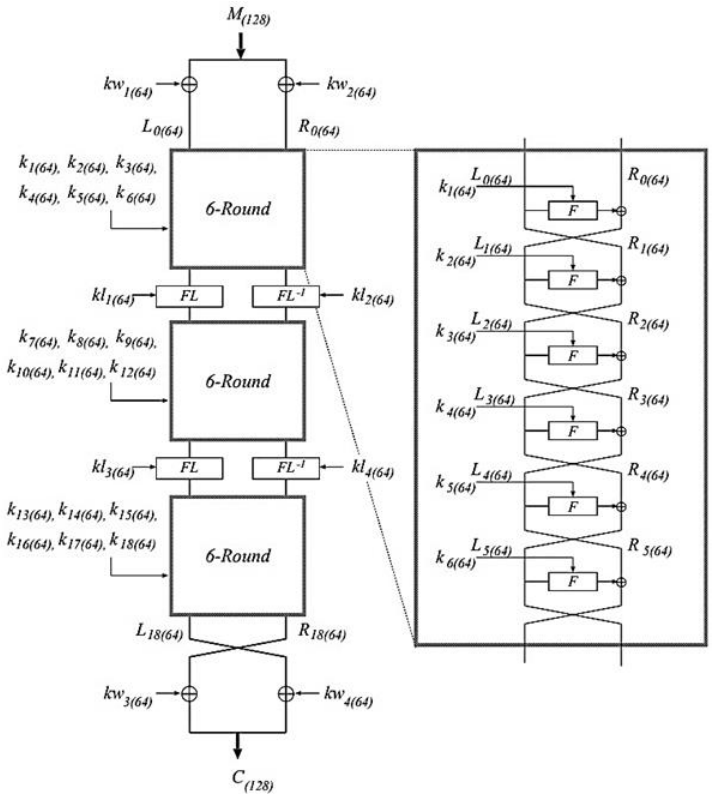

Fig. 2 Camellia encryption procedure using a 128-bit key[8]

Camellia key schedule:

The Camellia algorithm adopts a distinctive method in creating its key schedule. It divides the 128-bit key (k) equally into two parts (kl and kr); when using 128 bits, it assigns all bits to kl while kr retains a zero value. For a 192-bit key, kl takes the first 128 bits of k, and kr takes the remaining 64 bits; similarly, for a 256-bit key, where kl gets the first 128 bits, and kr gets the rest. These are used to generate other sub-keys (ka and kb). The formation of ka and kb is determined by specific six constant values (sigma1, sigma2, ..., sigma6). all these sub-keys are used to generate the round keys for Camellia through some left rotation when generating keys for encryption or right rotation when generating keys for decryption. The sub-key kb only concerns the 192-bit and 256-bit key domains[8]. The Camellia plant does not reveal its secrets casually, but it uncovers them through well-thought-out steps illustrated in an exquisite piece of art, as shown in Fig. 3, which displays the birth process of round keys.

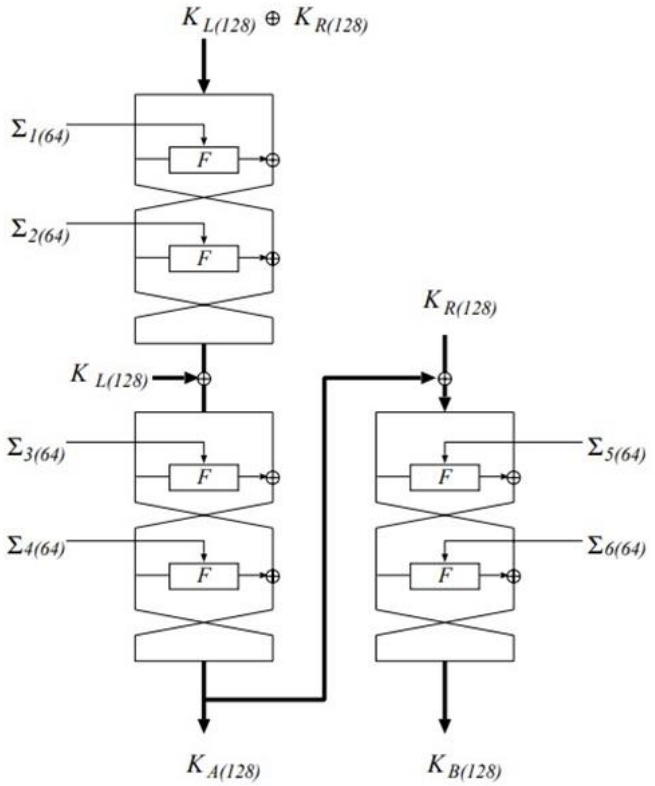

Fig. 3 Camellia key schedule[8]

Methodology:

Next, we will explain the method of producing round keys for AES using the Camellia key. Schedule. Our investigation specifically examined the 128-bit versions of both cryptographic. Techniques. For this study, we utilized the Camellia cryptographic engine, which was constructed using the Java programming language obtained from the official website of Nippon Telegraph and Telephone Corporation (NTT). The examination of this cryptographic technique found that the standard Camellia's F-Function generates 36 partial keys. More explicitly, it consists of eight additional keys for the FL and FL-1 functions and another eight for the weighting and post-weighting procedures. Each of these sub-keys is represented as a 32-bit integer value. In an effort to enhance analytical coherence, we merged the 36 sub-keys related to the F-Function with those designated for FL and FL-1 functions into one array, which led us to a total of 44 sub-keys. The composition of these sub-keys implies a sum total of 176 bytes (obtained by multiplying the number of sub-keys 44 by the number of bytes per sub-key 4) that matches the count of bytes in AES round keys. However, because AES requires its sub-keys to be organized in a matrix format, we had to convert the 44 sub-keys into a matrix with 4 rows and 44 columns. An exemplification of this transformation process is presented in **Fig. 4**. Following this methodological approach ensures that our implementation remains compatible with the AES framework. This will allow us to compare the cryptographic efficiencies of AES and what we achieve using the Camellia key schedule.

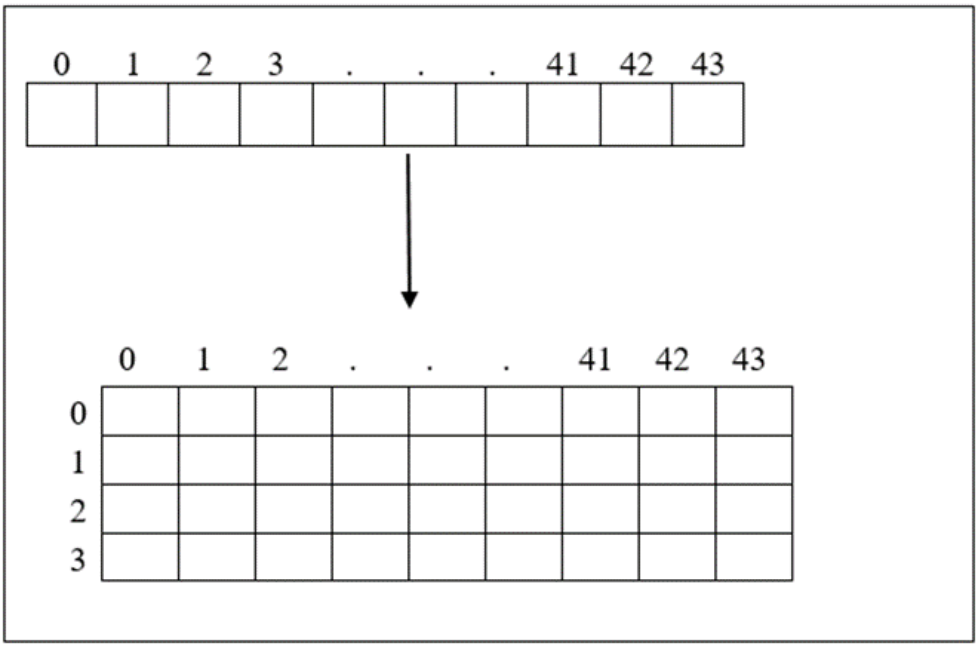

Fig. 4 Converting an array to a matrix.

Results and Discussion:

This portion of the investigation will demonstrate the results of the suggested enhancement to the algorithm. The NIST-800-22 statistical test suite evaluated the algorithm's effectiveness as a security tool. The suite is composed of 15 different statistical tests that each produce a unique numerical result called a P-value. The meaning of the value is interpreted to determine if the tested object meets or fails to meet each criterion using a pre-determinable threshold (often set at p-value≥0.01). This method allows us to determine the degree of randomness in our algorithm. During the evaluation, we contrasted AES to its original secret key and altered the key in Camellia in order to participate in the evaluation. To ensure the legitimacy of the conclusions, we attempted to utilize 50 different keys. The cryptographic outcomes derived from these keys were documented in an Excel file that was specifically designed to document and analyze the results. We can create a dataset with 50 values for each experiment, which was conducted in 16 different experiments (the cumulative sum experiment has two experiments, forward and backwards). The average of the 50 samples' calculations results in a single conclusion for each experiment, which describes the effectiveness of the algorithm in terms of the NIST criteria listed in **Table 1**, which includes the average value of both algorithms. This comprehensive analysis aims to explain in detail the enhanced safety properties of the algorithm. The findings are illustrated in **Fig. 5**, which shows the results of the test; our goal is for this extensive analysis to impact the field of cryptographic security significantly.

Table 1: Average values of test results for 50 keys

Test Name	AES+AES KEY	AES + Camellia KEY
Frequency Test	0.458071654	0.462435169
Frequency within a Block	0.517219693	0.576983698
Runs	0.491925618	0.518496196
Longest Run	0.516156822	0.560946249
Rank	0.504622502	0.556046434
Fourier Transform	0.530131231	0.530832171
Non-overlapping	0.503237877	0.528400923

Overlapping	0.457117939	0.562433577
Universal Test	0.540286887	0.487557292
Complexity	0.492120674	0.542757505
Serial Test	0.506499526	0.422453365
Entropy	0.480198879	0.464941376
Cumulative Sum Forward	0.453807934	0.500204197
Cumulative Sums backward	0.482722673	0.493993892
Excursions	0.490017423	0.528342318
Variants	0.502254133	0.525621519

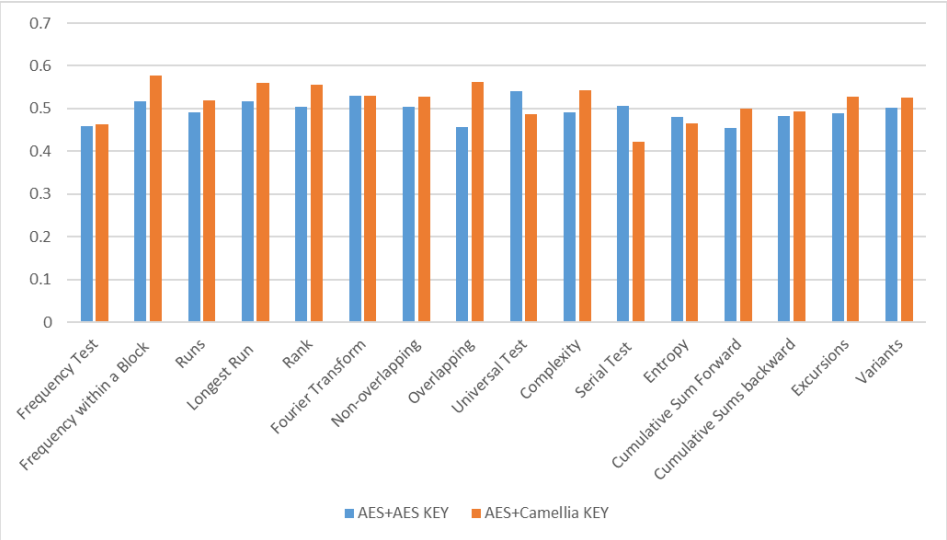

Fig. 5 Test results

The findings in **Table 1** indicate that all test results have successfully exceeded the predetermined P-value threshold. Furthermore, employing an Advanced Encryption Standard (AES) technique with a Camellia key, as depicted in **Fig. 5,** demonstrates substantial enhancements in feature performance, which diverges significantly from the results obtained utilizing the original key. This improvement enhances the effectiveness of the AES algorithm and highlights the significance of selecting the appropriate key to optimize algorithmic performance.

Conclusion:

In wrapping up, exploring the details and relationships present in cryptographic algorithms such as AES and Camellia can open doors to major breakthroughs in establishing encryption standards. Enhancing the security of vital information by implementing advanced cryptographic attacks based on securing data using key scheduling components from separate areas can be seen as an initiative from this research. We propose a thoughtful approach toward enhancing AES's key scheduling mechanism, which is expected to promote more academic interest and foster practical investigations. This work is hoped to inspire a generation of other works that would be both scholarly and practical in the field of cryptographic security, an area we see as having its development toward future enrichment.

References

1. NIST, D.E.S., *Advanced Encryption Standard (AES)(FIPS–197).* National Institute of Standards and Technology, 2001.
2. Derbez, P., et al. *Variants of the AES key schedule for better truncated differential bounds*. in *International Conference on Selected Areas in Cryptography*. 2018. Springer.
3. Partheeban, P. and V. Kavitha, *Dynamic key dependent AES S-box generation with optimized quality analysis.* Cluster Computing, 2019. **22**: p. 14731-14741.
4. Boura, C., P. Derbez, and M. Funk. *Alternative Key Schedules for the AES*. in *International Conference on Applied Cryptography and Network Security*. 2024. Springer.
5. Kato, A., S. Moriai, and M. Kanda, *The Camellia cipher algorithm and its use with IPSec*. 2005.
6. Hammood, M.M., T.S. Atia, and A.Y. Yousuf, *Design and Implement Pseudo Random Number Generator for Block Cipher Encryption Algorithm.* Iraqi Academic Scientific Journals, 2009. **14**(3): p. 13-16.
7. Khalil, A.A., A.M. Kaftan, and M.M. Hammoud, *Modify PRESENT Algorithm by New technique and key Generator by External unit.* Tikrit Journal of Pure Science, 2023. **28**(2): p. 97-103.
8. Aoki, K., et al. *Camellia: A 128-bit block cipher suitable for multiple platforms—design andanalysis*. in *Selected Areas in Cryptography: 7th Annual International Workshop, SAC 2000 Waterloo, Ontario, Canada, August 14–15, 2000 Proceedings 7*. 2001. Springer.
9. Wu, S. and M. Wang, *Security evaluation against differential cryptanalysis for block cipher structures.* Cryptology ePrint Archive, 2011.
10. Alsharida, R., et al. *RC4D: A New Development of RC4 Encryption Algorithm*. in *Selected Papers from the 12th International Networking Conference: INC 2020 12*. 2021. Springer.
11. Hammood, M.M. and K. Yoshigoe. *Previously overlooked bias signatures for RC4*. in *2016 4th International Symposium on Digital Forensic and Security (ISDFS)*. 2016. IEEE.
12. Pallavi, K., V.R. Kumar, and S. Srikrishna. *Comparative study of various lightweight cryptographic algorithms for data security between IoT and cloud*. in *2020 5th International Conference on Communication and Electronics Systems (ICCES)*. 2020. IEEE.
13. Daemen, J. and V. Rijmen. *AES and the wide trail design strategy*. in *Advances in Cryptology—EUROCRYPT 2002: International Conference on the Theory and Applications of Cryptographic Techniques Amsterdam, The Netherlands, April 28–May 2, 2002 Proceedings 21*. 2002. Springer.
14. Sachdeva, S. and A. Kakkar. *Implementation of AES-128 using multiple cipher keys*. in *Futuristic Trends in Network and Communication Technologies: First International Conference, FTNCT 2018, Solan, India, February 9–10, 2018, Revised Selected Papers 1*. 2019. Springer.
15. Zagi, H.R. and A.T. Maolood, *A novel serpent algorithm improvement by the key schedule increase security.* Tikrit Journal of Pure Science, 2020. **25**(6): p. 114-125.
16. Jat, D.S. and I.S. Gill. *Enhanced Advanced Encryption Standard with Randomised Round Keys*. in *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*. 2020. IEEE.
17. Cao, Z., et al. *Analysis and improvement of aes key expansion algorithm*. in *2022 International Conference on Artificial Intelligence and Computer Information Technology (AICIT)*. 2022. IEEE.
18. Shashankh, S., et al. *Affine recurrence based key scheduling algorithm for the advanced encryption standard*. in *Computer Networks and Inventive Communication Technologies: Proceedings of Fourth ICCNCT 2021*. 2022. Springer.
19. Zagi, H.R. and A.T. Maolood, *A New Key Generation to Greate Enhanced Security Version of AES Encryption Method.* Journal of College of Education, 2021(2).

20. Pub, N.F., *197: Advanced encryption standard (AES).* Federal information processing standards publication, 2001. **197**(441): p. 0311.
21. Daemen, J. and V. Rijmen, *The design of Rijndael*. Vol. 2. 2002: Springer.
22. Smid, M.E., *Development of the advanced encryption standard.* Journal of Research of the National Institute of Standards and Technology, 2021. **126**.
23. Aoki, K., et al., *Specification of Camellia-a 128-bit block cipher.* Specification Version, 2000. **2**.

IRAQI Academic Scientific Journals
Samarra Journal of Pure and Applied Science
SJPAS
مجلة سامراء للعلوم الصرفة والتطبيقية
www.sjpas.com
p ISSN: 2663-7405
e ISSN: 2789-6838

تعزيز أمان AES بناء على جدول مفاتيح كاميليا

أرشد سامي سليمان*، ميثم مصطفى حمود
قسم علوم الحاسوب، كلية علوم الحاسوب والرياضيات، جامعة تكريت، العراق

معلومات البحث:
تأريخ الاستلام: 2024/08/21
تأريخ التعديل: 2023/06/20
تأريخ القبـــول: 2023/07/20
تأريخ الـنشر: 2025/03/30
الكلمات المفتاحية:
الكلمة المفتاحية 1، الكلمة المفتاحية 2، الكلمة المفتاحية 3، الكلمة المفتاحية 4، الكلمة المفتاحية 5

معلومات المؤلف
الايميل:
الموبايل:

الخلاصة:

تتمثل الوظيفة الأساسية لمعيار التشفير المتقدم (AES) في الأمان؛ وهنا يأتي دور معيار التشفير المتقدم .(AES) إنّ الجمع بين معيار التشفير المتقدم (AES) والتطبيق له تأثير كبير على قدرته على حماية البيانات الحساسة. ويشمل ذلك الاتصالات الآمنة أو المعاملات المالية الخاصة. أدى اعتماد معيار التشفير المتقدم (AES)في عام 2001 إلى إجراء تقييمات واسعة النطاق أظهرت باستمرار فعالية معيار التشفير المتقدم في منع الهجمات من جميع الزوايا. يقترح التحقيق جعل الجدول الزمني لمفاتيح AES أكثر مرونة من أجل زيادة مقاومته لكل من الهجمات المباشرة والأخطار المحتملة الأخرى. ومن الأمور الأخرى التي يمكن أنْ يأخذها التحقيق بعين الاعتبار دمج بعض المعلومات من جدول كاميليا الأساسي للهجمات مع خوارزمية AES؛ وهذا من شأنه أنْ يؤدي إلى إعادة النظر في ضعف كاميليا أمام الهجمات الإلكترونية بسبب تصميمها المعقد. ومن ثم فإنّ دمج خصائص شيفرة كاميليا المتقدمة هذه في بنية AES القائمة بالفعل يمكن أنْ يؤتي ثمارًا غنية ووفيرة. بعد ذلك، استخدمنا مجموعة اختبارات NIST-800-22 لتقييم عشوائية الخوارزمية المعدلة، ثم قارنّا الخوارزمية الجديدة بخوارزمية AES الأصلية. أظهرت النتائج تحسينات كبيرة في الخوارزمية المحسّنة مقارنةً بخوارزمية AES القياسية.