Jurnal Nasional Teknik Elektro dan Teknologi Informasi
Volume 13 Number 3 August 2024

DOI: 10.22146/jnteti.v13i3.11824

Securing RFID in IoT Networks With Lightweight AES and ECDH Cryptography Approach

Robby Kurniawan Harahap[1], Alief Vickry Thaha Maulidzart[1], Antonius Irianto Sukowati[2], Dyah Nur'ainingsih[1], Widyastuti[1], Desy Kristyawati[1]

[1] Program Studi Teknik Elektro, Universitas Gunadarma, Depok, Jawa Barat 16424, Indonesia
[2] Program Studi Teknik Elektro, Universitas Cendekia Abditama, Tangerang, Banten 15810, Indonesia

[Received: 7 February 2024, Revised: 9 April 2024, Accepted: 5 July 2024]
Corresponding Author: Robby Kurniawan Harahap (email: robby_kurniawan@staff.gunadarma.ac.id)

ABSTRACT — Radio frequency identification (RFID) technology integrated into the Internet of things (IoT) networks often poses security and privacy concerns due to its attack vulnerability. This research proposed a lightweight cryptographic model tailored for implementation in resource-constrained environments. The objective is to address security challenges while accommodating limited memory, power, and size requirements. A combined modified 126-bit Advanced Encryption Standard (AES) algorithm with a 256-bit elliptic curve Diffie-Hellman (ECDH) cryptographic key was utilized to develop lightweight cryptography for securing RFID data. The implementation used the Python programming language in Jupyter Notebook, with RFID operating at 13.56 Mhz. The methodology involved retrieving RFID data through additional programs and equalizing ECDH keys for encryption and decryption. Encryption and decryption testing demonstrated a high success rate, achieving an accuracy of 99.9%. The first encryption attempt required 85.125 ms, with the second attempt completed faster at 65.95 ms, showcasing improved efficiency. File encryption sizes averaged 29.875 bytes for the initial attempt and 30.1 bytes for the subsequent one. This research was limited to algorithm evaluation and had not been implemented in hardware. However, the proposed hybrid cryptography offers significant benefits for maintaining the confidentiality of RFID data within IoT environments. Rapid, efficient, and compact encryption of unique identifier (UID) data ensures enhanced security, thereby addressing critical concerns associated with RFID-enabled IoT networks.

KEYWORDS — Advanced Encryption Standard (AES), Elliptic Curve Cryptography, IoT Networks, Lightweight Cryptographic, RFID Security.

I. INTRODUCTION

In the fast-paced digital landscape, radio frequency identification (RFID) technology plays a vital role in various sectors like supply chain management and access control. However, despite its numerous benefits, concerns arise regarding its susceptibility to cyber-attacks and potential threats to user privacy. RFID, a key component of automatic identification and data capture (AIDC), allows for the simultaneous reading of multiple RFID tags via wireless data transmission [1], [2], facilitating the detection, identification, tracking, and tracing of various objects [2]. When triggered by a compatible device's signal, an RFID tag identifies itself using an RFID reader [3]. Given RFID's role in transmitting information, ensuring information security is crucial, encompassing aspects like communication (encryption, authentication, and authorization), secure protocols, routing, and network security [4].

RFID technology has become integral to the Internet of things (IoT), fostering seamless connectivity among everyday objects and facilitating information exchange without human intervention. This collaboration not only enhances operational efficiency but also enables better understanding of specific applications, such as inventory tracking systems, by collecting and analyzing data [5]. However, securing RFID in IoT systems remains a significant challenge. Security keys, crucial for controlling encrypted operations like data encryption and decryption, play a pivotal role in determining the security level of a system or device. Lightweight encryption models, tailored for resource-constrained devices like RFID tags, are designed to operate efficiently in environments with limited resources such as memory, power, and size. Additionally, soft passwords must strike a balance between security, cost, and performance [6].

In the realm of IoT, characterized by high mobility and low storage limitations, execution speed and storage efficiency are paramount to support device functionality. Consequently, lightweight encryption becomes imperative. Research [7] has underscored the importance of security against attacks compromising IoT system infrastructure like supervisory control and data acquisition (SCADA) systems. Lightweight encryption algorithms aim to minimize computational load, ensuring fast encryption and decryption processes without compromising limited hardware resources. Furthermore, lightweight crypto can generate smaller encryption files, addressing the storage constraints of IoT devices and enabling efficient operation amidst mobility and resource limitations.

Previous research on lightweight cryptography in RFID devices utilizing Advanced Encryption Standard (AES) and elliptic curve Diffie-Hellman (ECDH), has shown growing interest in developing efficient and secure solutions for IoT networks [8]. Vulnerabilities of IoT devices to various attacks have emphasized the need for lightweight security protocols [9]. Lightweight cryptography, characterized by simplified encryption schemes with low computational complexity, offers a viable solution for resource-constrained devices like RFID tags [10]. While AES has been prevalent for secure communication among IoT devices, its limitations in highly constrained environments have spurred researchers to explore new optimized block ciphers [11]. Additionally, using ECDH

in conjunction with lightweight authentication mechanisms has been proposed to enhance IoT-constrained device security [12]. Overall, the research landscape indicates a shift towards developing lightweight cryptographic algorithms tailored to IoT devices' specific constraints and security requirements, particularly in RFID systems.

Research focusing on implementing cryptography for RFID devices highlights challenges in achieving the right balance between cost, performance, and security in lightweight cryptography [13]. Furthermore, limited computing and memory resources, especially in passive RFID technology contexts, present obstacles to RFID security systems in IoT [14]. These challenges underscore the need for further research on authentication, encryption, and security protocols to address limited resources hindering standard security functions such as AES encryption. Implementing traditional encryption methods can be challenging due to resource constraints, necessitating lightweight and ultra-lightweight authentication protocols suitable for low-cost RFID deployments. Applying the AES algorithm to RFID systems ensures the security of data transmission.

Despite existing constraints, opportunities remain for designing and implementing lightweight cryptography in RFID devices. This paper employs software and simulation methods to design and implement lightweight cryptography in RFID devices for IoT environments. The process involves integrating lightweight cryptographic algorithms into hardware and software for RFID tags [15]. Targeted outcomes focus on average accuracy, speed, and significant data size, with the research expected to make valuable contributions to implementing cryptography as a secure and high-quality RFID device in the IoT environment.

The paper is structured as follows: Section II, Material and Method, discusses the concept of lightweight cryptography and the methods employed. Section III presents the results achieved through simulations and testing. A discussion on the results and a comparison with existing literature are provided in Section IV. Finally, the paper concludes by summarizing the findings and their implications.

II. MATERIAL AND METHODOLOGY

A. IOT OVERVIEW

The IoT has ushered in a new era of research dominance, with applications spanning smart logistics and transportation, smart health, smart environment, smart infrastructure (including smart cities, homes, offices, malls, and Industry 4.0), smart agriculture, and more. IoT constitutes a network of interconnected objects, each with unique codes facilitating data collection and sharing over the Internet, autonomously or with human interaction [13]. IoT devices can be broadly categorized into two groups: those with abundant resources, such as servers, personal computers, tablets, and smartphones; and those with limited resources, such as industrial sensors, sensor nodes, RFID tags, and actuators.

Across diverse domains, numerous IoT implementations have been conducted, including the maritime IoT (MIoT), which entails a network of interconnected devices, sensors, and systems utilized in the maritime industry for applications like ship tracking, logistics, supply chain management, and maritime operations [14]. Combining IoT and RFID technologies enhances vehicle security and assists in preventing theft in vehicle security systems [16]. A primary

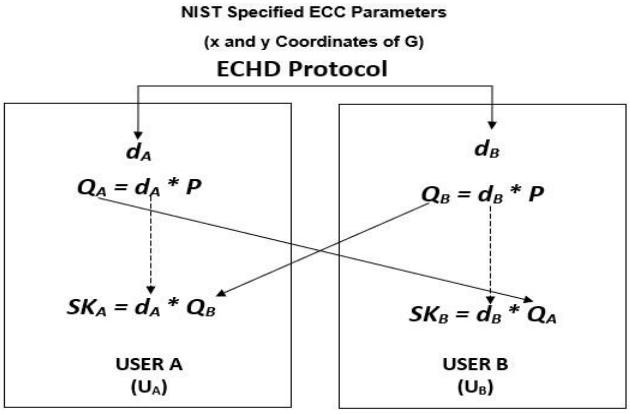

Figure 1. ECDH protocol concept.

focus of IoT revolves around monitoring activities, such as animal feeding monitoring [17]. However, cybersecurity poses a significant challenge in IoT systems research, encompassing confidentiality, data integrity, authentication, and authorization. The IoT security systems frequently utilize cryptographic approaches to handle these cybersecurity aspects.

B. LIGHTWEIGHT CRYPTOGRAPHY

Lightweight cryptography plays a vital role in ensuring secure communications, particularly for resource-constrained devices like those found in the IoT. As identified in [13], research challenges in IoT security include limited memory (registers, RAM, ROM), reduced computing power, small physical assembly areas, low battery power, and the need for real-time responsiveness. These limitations stem from the small size and limited resources typical of IoT devices. Conventional cryptographic standards, when applied to IoT devices, especially in real-time applications like RFID, often struggle to provide fast and accurate responses with critical security using available resources. However, lightweight cryptography offers a solution by leveraging features such as small memory footprint, low processing power, minimal energy consumption, and real-time responsiveness, even in devices with limited resources.

In IoT, lightweight cryptographic primitives encompass four types: lightweight block cipher (LWBC), lightweight stream ciphers (LWSC), lightweight hash functions (LWHF), and elliptic curve cryptography (ECC) [18]. ECC, a newer form of public key cryptography, offers quick key agreements, signatures, and key generation, although it does not explicitly provide encryption mechanisms. Nevertheless, ECC boasts advantages such as minimal memory usage, reduced energy consumption, optimized power factor, and improved speed.

AES, renowned for its high security level, stands as a popular encryption method due to its effectiveness, simplicity, and broad platform support. Offering key lengths of 128, 192, and 256 bits [8], AES serves as a versatile choice for encryption. However, this study utilized 128-bit keys and 16 looping processes. Each encryption and decryption process requires a key, which can be a word or a phrase, and these keys are integral to cryptographic methods for data security. Adopted as the Federal Information Processing Standard (FIPS) by the National Institute of Standards and Technology (NIST) in 2001 [19], AES encryption is widely recognized for its reliability. The ECDH protocol facilitates a key agreement scheme, enabling parties A and B to establish a shared secret key for private key algorithms. Through the exchange of public

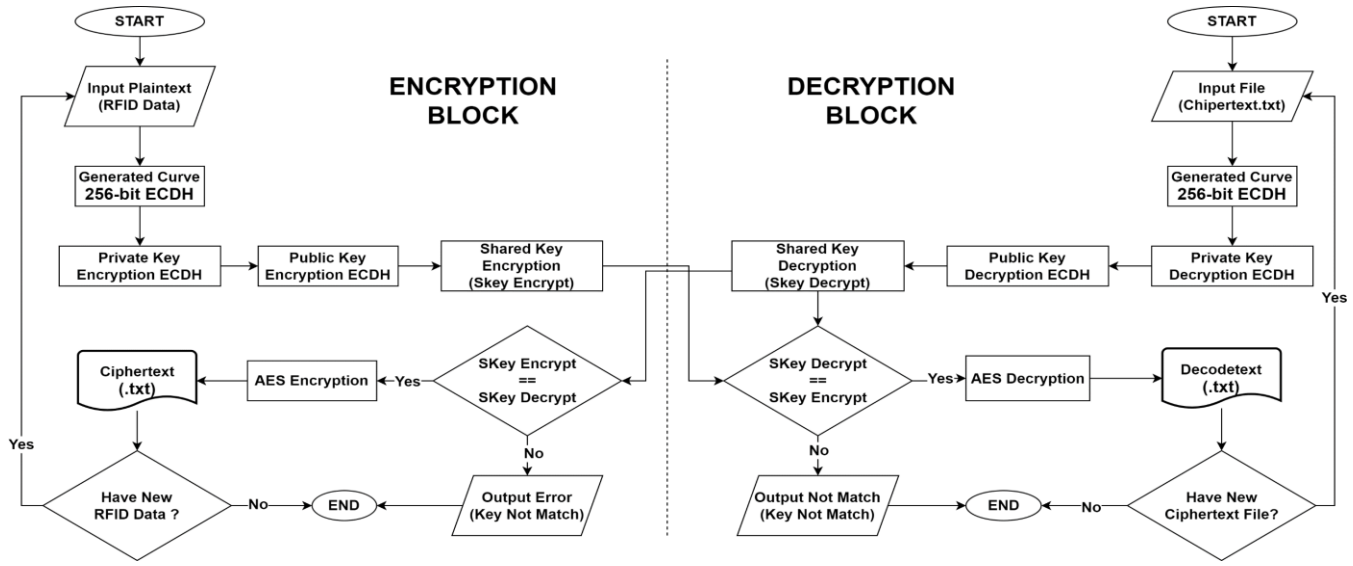

Figure 2. AES and ECDH lightweight cryptograph flowchart.

information, both parties can generate a shared secret key using their respective public and private information. Notably, third parties cannot deduce the shared secret key from publicly available information without knowledge of the parties' personal details [20]. Research [21] conducted a simple application of lightweight cryptography, implementing a lightweight cipher on application-specific integrated circuit (ASIC) and field-programmable gate array (FPGA) devices to optimize energy usage.

An ECC variant of the Diffie–Hellman protocol is the ECDH, designed for key agreement (or shared key generation) between two users. Prior research has highlighted that the standard of the ECDH algorithm is vulnerable to man-in-the-middle attacks, where the attacker can read and modify all sent messages without targeting legitimate users [22], [23]. Two solutions are proposed to strengthen the ECDH algorithm against these attacks. First, user public key authentication ensures the validation of the user's static public key. Second, both parties generate temporary public keys for each communication session. This approach enables perfect forward secrecy (PFS) and reduces algorithmic complexity, eliminating the need for additional authentication calculations. Cryptographic computing for each layer in Figure 1 involves several algorithms or protocols [24], [25]. Each user U_A and U_B initiates shared-key generation using standard ECC parameters. Subsequently, each user U_A and U_B generates its public key using the base point generator (G) and the private key. In [26], the algorithm for the ECDC key exchange was examined and evaluated using the PyCryptodome libraries and the elliptic curve integrated encryption scheme.

C. METHOD

This research employed the concept of hybrid cryptography introduced in [27], combining modifications of the AES algorithm with ECDH. This approach aimed to develop lightweight cryptography to uphold RFID data security. The encryption and decryption processes utilized AES as a symmetric cryptography with key lengths of 128 bits, 192 bits, and 256 bits and a packet size of 128 bits, rendering the AES encryption system highly adaptable. Consequently, AES encryption technology is widely implemented in hardware and software, while ECDH serves as asymmetric cryptography [28]. The ECDH, generates keys using elliptic curves [29]. Under the master agreement scheme, all involved parties in a particular communication must contribute data or information to generate shared session keys [30]. This study employed the Python programming language via Jupyter Notebook in Anaconda Navigator, utilizing two laptops, with one laptop dedicated to testing the encryption or decryption block. A hybrid encryption scheme with an ECDH key exchange scheme was employed.

Figure 2 illustrates the method for encryption and decryption using AES with ECDH keys. The method comprises two blocks: the encryption block and the decryption block. The initial step was generating the private key for each block using the "brainpoolP256r1" curve 256-bit library, followed by generating the private key. The resultant private key in the encryption block was stored for the encryption ECDH public key and for the AES decryption process in the decryption block, and vice versa. The private key results in the decryption block were stored for the decryption ECDH public key and the AES encryption process in the encryption block. According to the rules of ECDH, if two secret numbers, "a" and "b" (representing the private keys of U_A and U_B), are combined with an ECC elliptic curve featuring a generator point (G), the U_A PrivKey G and UB PrivKey G, values could be exchanged through an insecure channel. Thus, U_A and U_B public keys are utilized in (1).

$$U_A\ PubKey \times U_B PrivKey = U_B PubKey \times U_A PrivKey. \quad (1)$$

The AES encryption process involved modifying the standard procedure by incorporating an ECDH key. Each round of encryption comprised four functional steps: SubBytes, ShiftRows, MixColumns, and AddRoundKey. These steps operated on standard blocks of AES data, with each block represented by a matrix containing 16 or 128 bits. Upon successful encryption, the results were stored in a file. Similarly, the hybrid encryption process utilizing the ECDH key produced encrypted data, which were also saved in a file. Subsequently, an analysis stage ensued, wherein the encrypted output (ciphertext) and decrypted output were stored in .txt format. This analysis was conducted to evaluate parameters such as encryption process time, file sizes, and runtime.

Integrating AES and ECDH keys within a hybrid cryptographic framework presents a robust solution to address

Figure 3. Research method.

TABLE I
RESULTS OF RFID UID DATA COLLECTION OF THE FIRST EXPERIMENT

RFID ID	UID RFID (Plaintext)
RFID Card 1	638FE118
RFID Card 2	1499C649
RFID Card 3	A8972327
RFID Card 4	83139205
RFID Card 5	E3670B04
RFID Key 1	A7C6C128
RFID Key 2	4C832949
RFID Key 3	B61D62AF

TABLE II
RESULTS OF RFID UID DATA COLLECTION OF THE SECOND EXPERIMENT

RFID ID	UID RFID (Plaintext)
RFID Card 6	F3F2BADF
RFID Card 7	0DD60D04
RFID Card 8	AB95B9DF
RFID Card 9	41C755FF
RFID Card 10	979ABBDF
RFID Card 11	271ABADF
RFID Card 12	D9CDBBDF
RFID Card 13	A7E6B8DF
RFID Card 14	00A8BADF
RFID Card 15	22F7BBDF

RFID data security concerns, drawing upon the advantages of both symmetric and asymmetric cryptography. The method established secure communication channels by incorporating ECDH key exchange during encryption, reducing the likelihood of unauthorized access or data interception. Moreover, rigorous testing and analysis, encompassing assessments of encryption process duration, file sizes, and runtime comparisons, affirmed the method's dependability and efficiency in bolstering security measures while sustaining operational effectiveness. This comprehensive evaluation, coupled with the detailed depiction of the research methodology in Figure 3, furnished compelling validation of the efficacy and integrity of the proposed approach.

III. RESULTS

The exploration into hybrid cryptographic algorithms, combining AES with ECDH keys, progressed to the results phase. At this juncture, the research findings are presented, elucidating various parameters, including RFID unique identifier (UID) data collection, outcomes of encryption and decryption tests, processing times for encryption and decryption, as well as the sizes of the original plaintext file, encrypted ciphertext file, and resulting decodetext file. Additionally, a comparative analysis of processing times and file sizes was conducted. This study undertook two distinct experiments, each employing different ECDH keys. The initial experiment involved eight UID RFID items (5 RFID cards and 3 RFID keychains), while the second experiment utilized ten UID RFID cards.

TABLE III
RESULTS OF ENCRYPTION TESTING IN FIRST EXPERIMENT

RFID ID	Ciphertext	Encryption Time (ms)
RFID Card 1	Encrypting... ChiperText : [VòØõÿðüË▯▯Ï▯ËÊ▯	86
RFID Card 2	Encrypting... ChiperText : öÕðü ▯ø▯▯▯▯▯▯▯È▯▯	95
RFID Card 3	Encrypting... ChiperText : Ü½¨▯▯E▯▯ÏÈ▯▯▯▯Ï▯	106
RFID Card 4	ChiperText : ÒV§▯▯▯©ÜÏÈ▯▯▯▯Ï▯ Encrypting...	98
RFID Card 5	Encrypting... ChiperText : ¯V ▯G▯▯ÇÏÈ▯▯▯▯Ï▯	101
RFID Key 1	Encrypting... ChiperText : ÜÀÃRðEü▯ÏÈ▯▯▯▯Ï▯	85
RFID Key 2	Encrypting... ChiperText : ▯Ú¤▯ÿÎð▯ÏÈ▯▯▯▯Ï▯	94
RFID Key 3	Encrypting... ChiperText : A▯§▯▯▯▯^ÏÈ▯▯▯▯Ï▯	61
Total encryption time		726

A. RFID DATA COLLECTION

RFID data collection involved gathering UID data from RFID chips, which was then utilized as plaintext for the encryption process. The collected data were stored in files in .txt format. The first experiment, as detailed in Table I, utilized 8 RFID tags comprising 5 RFID cards and 3 RFID keys. Each tag was assigned a distinct eight-digit UID, serving as a unique identifier for data readers. Similarly, the subsequent experiment, outlined in Table II, involved ten RFID cards, maintaining the same tag and UID characteristics as the initial experiment.

B. TESTING ENCRYPTION RESULTS AND ENCRYPTION RUNTIME

During encryption testing, the RFID UID data (plaintext) was transformed into unreadable characters, known as ciphertext. This study employed a 256-bit curve for ECDH keys and a 128-bit AES. Subsequently, the encryption process time was measured to determine the duration required to convert from plaintext to ciphertext. The outcome for the encryption key (ECDH) was 69783146250579664957949920515203566587300269878233 818733998581244206265167833, representing the ECDH key utilized in the encryption process during the initial attempt. Any discrepancy between the generated ECDH key used for encryption and decryption could result in variations in the plaintext.

Table III presents the outcomes of the initial hybrid encryption trial conducted on eight RFIDs. It is evident that encryption yielded a series of random characters, unreadable by humans. All RFID UID data were successfully encrypted

TABLE IV
RESULTS OF ENCRYPTION TESTING IN THE SECOND EXPERIMENT

RFID ID	Ciphertext	Encryption Time (ms)
RFID Card 6	Encrypting... ChiperText : ▯V▯▯Ù▯▯F▯▯▯É▯▯▯È	65
RFID Card 7	Encrypting... ChiperText : Ã×▯X▯Æ▯Â▯▯▯É▯▯▯È	41
RFID Card 8	Encrypting... ChiperText : Ü þ£ü▯▯F▯▯▯É▯▯▯È	60
RFID Card 9	Encrypting... ChiperText : ▯¥▯▯¥▯▯§▯▯▯É▯▯▯È	69
RFID Card 10	Encrypting... ChiperText : ùÀþßD▯▯F▯▯▯É▯▯▯È	75
RFID Card 11	Encrypting... ChiperText : ▯AñßÙ▯▯F▯▯▯É▯▯▯È	70
RFID Card 12	Encrypting... ChiperText : ▯±▯▯D▯▯F▯▯▯É▯▯▯È	70
RFID Card 13	Encrypting... ChiperText : ÙÀ¨Xδ▯▯F▯▯▯É▯▯▯È	80
RFID Card 14	Encrypting... ChiperText : Ã▯Ùöú▯▯F▯▯▯É▯▯▯È	59
RFID Card 15	Encrypting... ChiperText : ▯^▯▯D▯▯F▯▯▯Ê▯▯▯È	80
Total encryption time		669

without encountering any errors. In the first experiment, the fastest encryption process occurred in 85 ms, attributed to the encryption of RFID Key 3 with UID B61D62AF. Conversely, the most prolonged encryption process lasted 106 ms, associated with the encryption of RFID Card 3 with UID A8972327. The cumulative encryption time in the first attempt amounted to 726 ms. The encryption key (ECDH) outcome was 58063897800752943436263595813322921895668110925270326760817949421271248100 27, representing the ECDH key utilized in the encryption process during the subsequent attempt. Any discrepancy between the generated ECDH key used for encryption and decryption could resulted in variations in the plaintext.

Table IV shows the outcomes of the initial hybrid encryption endeavor conducted on ten RFIDs. It is evident that encryption generates a sequence of random characters, indiscernible to humans. All RFID UID data were effectively encrypted without encountering any errors. In the second experiment, the swiftest encryption process lasted 41 ms, attributed to the encryption of RFID Card 7 with UID 0DD60D04. Conversely, the lengthiest encryption process extended to 80 ms, involving RFID Card 13 with UID A7E6B8DF and RFID Card 15 with UID 22F7BBDF. The cumulative encryption time in the initial attempt totaled 669 ms.

C. TESTING DECRYPTION RESULTS AND DECRYPTION RUNTIME

The decryption test transforms ciphertext files comprising unreadable characters into decodetext, making them legible to humans. Employing a 256-bit curve for ECDH keys and a 128-bit AES, the decryption process aimed to ascertain the time required to convert ciphertext to decodetext. The file underwent decryption by inputting the encrypted .txt file, with the resultant text saved in the .txt format. The decryption key (ECDH) used in the first experiment was 48821956027739608857925916805825085142072912266785442189318763110805727647718. Should the ECDH key not align with the encryption's key generation, the decrypted text would differ from the plaintext. Table V exhibits the status of

TABLE V
RESULTS OF DECRYPTION TESTING IN THE FIRST EXPERIMENT

RFID ID	Decryption File Name	Decryption Time (ms)
RFID Card 1	Deskrip 29-07-2023(40).txt	68
RFID Card 2	Deskrip 29-07-2023(24).txt	72
RFID Card 3	Deskrip 29-07-2023(09).txt	81
RFID Card 4	Deskrip 29-07-2023(17).txt	86
RFID Card 5	Deskrip 29-07-2023(23).txt	85
RFID Key 1	Deskrip 29-07-2023(31).txt	90
RFID Key 2	Deskrip 29-07-2023(39).txt	88
RFID Key 3	Deskrip 29-07-2023(36).txt	66
Total overall decryption runtime		636

successful decryption time in the first experiment. The swiftest decryption time was recorded for RFID Key 3 at 66 ms, while RFID Key 1 had the lengthiest decryption time of 90 ms. The cumulative decryption processing time in the initial attempt totaled 636 ms. The decryption process automatically saves the resultant text in the decrypted [date of decryption].txt format.

The ECDH key utilized for the decryption process in the second experiment was represented by the value of 68723788531881136467441998490530808783086550182713890595138556198550535739 96. It is crucial for this key to match the one generated during encryption; otherwise, disparities between the ciphertext and plaintext may arise. Table VI provides insights into the decryption test times during the second experiment. Notably, RFID Card 11 demonstrated the fastest decryption time, completing the process in 40 ms; whereas, RFID Card 9 exhibited the lengthiest decryption time, taking 101 ms. The cumulative decryption processing time for the second experiment totaled 650 ms.

D. MEASUREMENT RESULTS OF ORIGINAL FILE SIZE, ENCRYPTED FILES AND DECRYPTED FILES

The measurement results serve to determine the sizes of the original file (plaintext), encryption result file (ciphertext), and decryption result file (decodetext). Table VII showcases the sizes of the plaintext, ciphertext, and decodetext files from the initial attempt. The smallest plaintext file size recorded was 7 bytes, observed in three RFID files. Meanwhile, the largest was 8 bytes, found in five RFID files, resulting in an average total size of 7.625 bytes. Regarding encryption (ciphertext) file sizes, RFID Card 2 and RFID Card 3 had the largest files at 31 bytes each, whereas RFID Key 3 had the smallest at 28 bytes, with an average size of 29.875 bytes. Additionally, the size of the decodetext for all RFID files was consistently 16 bytes. Upon summing the sizes of all files processed, the total size was 61 bytes for the original file, 269 bytes for the ciphertext file, and 128 bytes for the decoded text.

Table VIII displays the sizes of the plaintext file, ciphertext file, and decodetext resulting from the initial attempt. The smallest plaintext file measured 7 bytes, noted in five RFID files, while the largest, at 8 bytes, was observed in the same number of RFID files, resulting in an average total size of 7.5 bytes. Regarding ciphertext file sizes, RFID Card 9 had the largest file at 32 bytes, while RFID Card 15 had the smallest at 28 bytes, averaging 30.1 bytes. Furthermore, the size of the decodetext for all RFID files remained consistent at 16 bytes. From the entire process, the total size was 75 bytes for the

TABLE VI
RESULTS OF DECRYPTION TESTING IN THE SECOND EXPERIMENT

RFID ID	Decryption File Name	Decryption Time (ms)
RFID Card 6	Deskrip 30-07-2023(33).txt	60
RFID Card 7	Deskrip 30-07-2023(52).txt	50
RFID Card 8	Deskrip 30-07-2023(22).txt	71
RFID Card 9	Deskrip 30-07-2023(33).txt	101
RFID Card 10	Deskrip 30-07-2023(00).txt	75
RFID Card 11	Deskrip 30-07-2023(18).txt	40
RFID Card 12	Deskrip 30-07-2023(26).txt	55
RFID Card 13	Deskrip 30-07-2023(25).txt	65
RFID Card 14	Deskrip 30-07-2023(11).txt	60
RFID Card 15	Deskrip 30-07-2023(41).txt	73
Total overall decryption runtime		650

original file, 301 bytes for the ciphertext file, and 160 bytes for the decodetext.

IV. DISCUSSION

A. COMPARISON OF ENCRYPTION TIME AND DESCRIPTION TIME

The comparison between the processing time of hybrid encryption and hybrid decryption is a crucial aspect of evaluating the efficiency of the cryptographic methods utilized. This analysis examined the time required for encryption and decryption processes, shedding light on the intricacies of transforming RFID UIDs. Figure 4 vividly portrays the temporal contrast between the encryption and decryption results from the initial attempt. The data showed that the encryption process typically took longer than decryption, as indicated by the blue and red bar lines.

Figure 4 provides valuable insights into the temporal dynamics of encryption and decryption. It highlights the temporal gap between the two operations during the initial attempt, with encryption generally taking longer than decryption. Further analysis revealed an average encryption time of 90.75 ms and an average decryption time of 79.5 ms.

The average processing times derived from this analysis offer quantifiable evidence of the temporal differences inherent in hybrid encryption and decryption. With encryption averaging 90.75 ms and decryption 79.5 ms, it is apparent that decryption is generally faster. It underscores the importance of temporal considerations in assessing cryptographic techniques and suggests potential optimizations to enhance system performance. Figure 5 visually represents the temporal contrast between encryption and decryption in the second attempt. The blue bar line depicts the encryption period, while the red bar line illustrates the decryption time. Unlike the initial attempt, this iteration's time gaps between encryption and decryption are less pronounced; it can be stated that timing between encryption and decryption is more evenly distributed. This result suggests that there are potential efficiency gains in cryptographic operations. There is a slight difference between the two procedures, with an average encryption time of 66.9 ms and an average decryption time of 65 ms. The convergence of processing durations indicates opportunities for refining cryptographic methods, leading to more balanced and efficient encryption-decryption processes.

TABLE VII
RESULTS OF ORIGINAL FILE SIZE, ENCRYPTED FILES, AND DECRYPTED FILES IN THE FIRST EXPERIMENT

RFID ID	Original File Size (Bytes)	Ciphertext File Size (Bytes)	Decodetext File Size (Bytes)
RFID Card 1	7	30	16
RFID Card 2	8	31	16
RFID Card 3	8	31	16
RFID Card 4	8	29	16
RFID Card 5	8	30	16
RFID Key 1	7	30	16
RFID Key 2	7	30	16
RFID Key 3	8	28	16
Total size	61	269	128
Average	7.625	29.875	16

TABLE VIII
RESULTS OF ORIGINAL FILE SIZE, ENCRYPTED FILES, AND DECRYPTED FILES IN THE SECOND EXPERIMENT

RFID ID	Original File Size (Bytes)	Ciphertext File Size (Bytes)	Decodetext File Size (Bytes)
RFID Card 6	7	29	16
RFID Card 7	8	31	16
RFID Card 8	8	30	16
RFID Card 9	7	32	16
RFID Card 10	8	30	16
RFID Card 11	7	30	16
RFID Card 12	7	30	16
RFID Card 13	8	30	16
RFID Card 14	7	31	16
RFID Card 15	8	28	16
Total size	75	301	160
Average	7.5	30.1	16

B. FILE SIZE COMPARISON BETWEEN PLAINTEXT, CHIPERTEXT, AND DECODETEXT

The comparative test among plaintext, ciphertext, and decodetext serves as a crucial evaluation of file sizes, shedding light on the efficiency and effectiveness of cryptographic procedures. This study scrutinized the relative dimensions of the initial file, the encrypted file (ciphertext), and the resulting decoded text (decodetext). Figure 6 provides a detailed illustration of the size comparison, with the blue, red, and green bars representing the original, encrypted, and decodetext files, respectively.

Upon analyzing the size comparison depicted in Figure 6, discernible patterns emerged in the file sizes of the original, encrypted, and decoded files. The encrypted file exhibited a considerable increase in size due to the utilization of complex and random characters in the ciphertext, rendering it incomprehensible to humans. In contrast, the original file demonstrated the smallest size among the three variables. This divergence underscores the significant impact of encryption on file size, emphasizing the trade-off between information security and data volume.

The second trial yielded a quantitative analysis indicating an average discrepancy of 22.25 bytes between the initial and

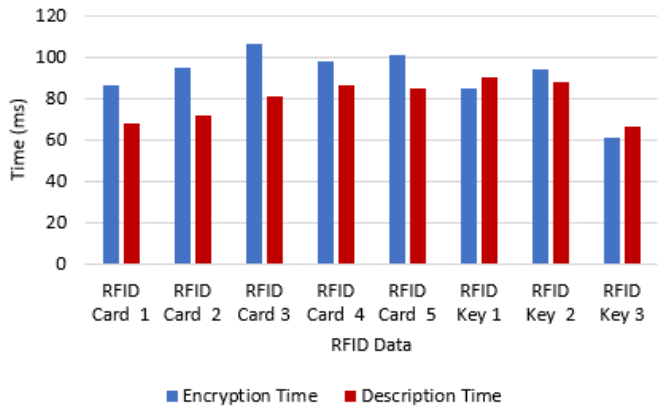

Figure 4. Comparison of encryption time and description time in the first experiment.

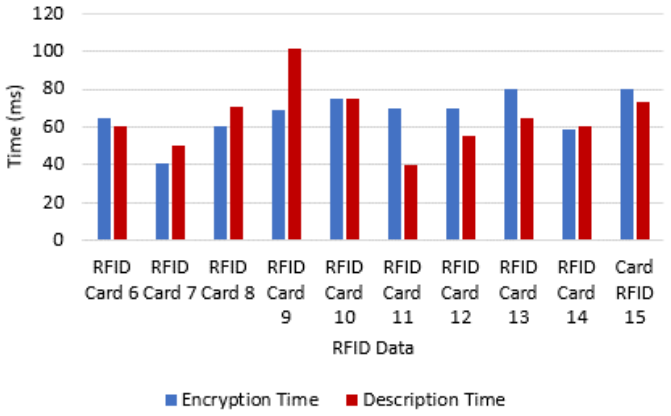

Figure 5. Comparison of encryption time and description time in the second experiment.

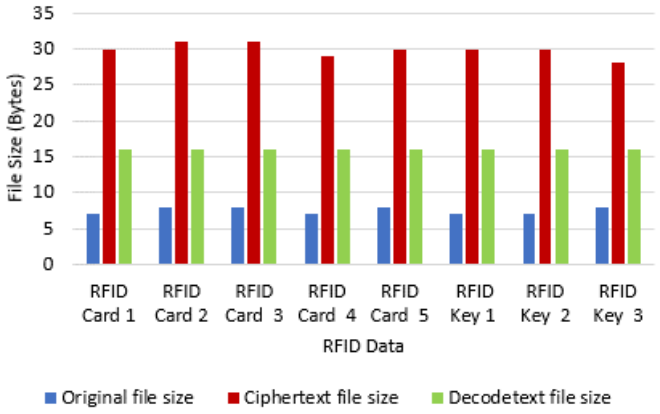

Figure 6. Size difference between plaintext, ciphertext, and decodetext for the first experiment.

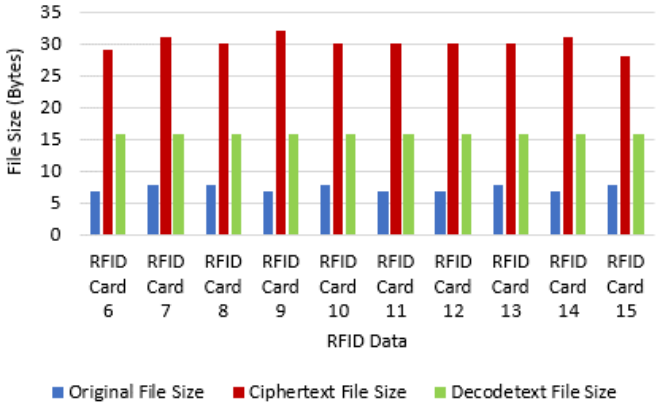

Figure 7. Size difference between plaintext, ciphertext, and decodetext for the second experiment.

encrypted file sizes. This variance underscores the influence of encryption on data volume and the importance of considering file sizes in cryptographic processes. The results of this comparative test provide significant insights for optimizing cryptographic protocols to balance security requirements and practical factors, such as limitations on file size.

Figure 7 provides a comprehensive analysis of file sizes, encompassing the original file, encrypted file (ciphertext), and resulting decoded text file (decodetext) from the initial attempt. Each bar in the graph represents a file's size, with the blue bar indicating the original file, the red bar representing the encrypted file, and the green bar representing the decoded text file. The encrypted file exhibited a significant size increase due to the presence of intricate and randomized characters, making it impossible for humans to understand. In addition, it underscores the impact of the cryptography procedure on data volume, as the incorporation of intricate characters contributes to its enlarged size. Conversely, the original file maintained the smallest size among the three variables, highlighting the substantial impact of encryption on file dimensions. Hence, it served as the standard measurement for comparison. Furthermore, the decryption procedure resulted in a file size representing the connection between the original and encrypted files, functioning as a physical representation of the cryptographic transformation.

Upon additional investigation in the context of the second experiment, it was found that there was an average discrepancy of 22.6 bytes between the original file and the size of the encrypted file. This discovery emphasizes the persistent influence of encryption on the amount of data, regardless of how often it is done, thus highlighting the significance of taking file sizes into account in cryptography efforts. Furthermore, it emphasizes the importance of enhancing cryptographic protocols to achieve a harmonious equilibrium between security prerequisites and practical limitations associated with file size considerations.

Several key aspects can be compared between the research findings of this study and the study [8] to ascertain the effectiveness and superiority of each technique. First, regarding the objective, while both studies aimed to enhance cryptographic techniques, this research explicitly targeted lightweight cryptography for IoT networks. This focus directly responds to the increasing demand for secure IoT implementations within resource-constrained environments, suggesting a more tailored and relevant solution compared to the broader objective of enhancing AES strength [8].

Second, in terms of approach and techniques used, this research introduced a novel hybrid cryptographic approach by combining modified AES with ECDH keys tailored explicitly for IoT environments. In contrast, [8] focused on modifying AES's SubBytes and ShiftRows transformations to enhance its strength. While both approaches exhibit innovation, the hybrid approach in this research demonstrates a more holistic solution by addressing IoT networks' unique constraints and requirements, potentially offering better adaptability and efficiency in real-world implementations.

Finally, it is evident from the performance metrics in Table I that this research achieved a high success rate of 99.9% in

encryption and decryption processes, with encryption times ranging from 85.125 ms to 65.95 ms (Table I). Although [8] reported an avalanche effect of 57.81% for the modified AES algorithm, the performance comparison regarding encryption times suggested that this research offered faster processing speeds. Additionally, the file encryption sizes in both studies are comparable, indicating similar efficiency in data handling. Therefore, based on these aspects, this research presents a more tailored, efficient, and potentially superior solution for securing IoT networks compared to the findings from research [8].

V. CONCLUSION

In summary, the evaluation of encryption and decryption procedures yielded promising outcomes in advancing lightweight cryptography. The efficacy of employing AES and ECDH keys is apparent, with an impressive success rate of 99.9% observed in both encryption and decryption tests. The reduction in processing time from the initial attempt, which required 85.125 ms, to the subsequent iteration, which required 65.95 ms, signifies enhanced efficiency. Despite manual encryption and decryption methodologies being utilized, the findings underscore the potential of lightweight cryptography for IoT networks, offering rapid data retrieval and compact file sizes.

In the future, there is a significant opportunity to augment this study by implementing lightweight cryptography on hardware platforms, particularly by leveraging microcontrollers. Integrating hardware-based implementation could substantially enhance the efficiency and applicability of cryptography algorithms in practical scenarios. Furthermore, future research endeavors could explore avenues to optimize processing times' efficiency and minimize file sizes. Researchers can delve deeper into lightweight cryptography and its relevance in emerging technological landscapes, leveraging the insights gleaned from this study. Such advancements will ultimately enhance the secure and efficient data transmission within IoT networks.

CONFLICTS OF INTEREST

The authors declare no conflicts of interest.

AUTHORS' CONTRIBUTIONS

Conceptualization, Robby Kurniawan Harahap; methodology, Robby Kurniawan Harahap; software, Alief Vickry Thaha Maulidzart; writing—original draft preparation, Antonius Irianto Sukowati and Dyah Nur'ainingsih; writing—reviewing and editing Widyastuti and Desy Kristyawati.

REFERENCES

[1] A. Haibi *et al.,* "Systematic mapping study on RFID technology," *IEEE Access,* vol. 10, pp. 6363–6380, Jan. 2022, doi: 10.1109/ACCESS.2022.3140475.

[2] W.C. Tan and M.S. Sidhu, "Review of RFID and IoT integration in supply chain management," *Oper. Res. Perspect.,* vol. 9, pp. 1–17, Feb. 2022, doi: 10.1016/j.orp.2022.100229.

[3] R. Hassan, A.A. Majeed, and Muqorobin, "Fingerprint data security system using AES algorithm on radio frequency identification (RFID) based population system," *Int. J. Inf. Technol. (INJIT),* vol. 1, no. 1, pp. 14–20, Jan.-Apr. 2023.

[4] J.R. Naif, G.H. Abdul-Majeed, and A.K. Farhan, "Secure IoT system based on chaos-modified lightweight AES," in *2019 Int. Conf. Adv. Sci. Eng. (ICOASE),* 2019, pp. 1–6, doi: 10.1109/ICOASE.2019.8723807.

[5] E.B. Setiawan, A. Yunita, and S.R. Sekarjatiningrum, "Development of automatic real time inventory monitoring system using RFID technology in warehouse," *JOIV, Int. J. Inform. Vis.*, vol. 6, no. 3, pp. 636–642, Sep. 2022, doi: 10.30630/joiv.6.3.1231.

[6] S.Q.A. Al-Rahman, A.M. Sagheer, and O.A. Dawood, "NVLC: New variant lightweight cryptography algorithm for Internet of things," in *2018 1st Annu. Int. Conf. Inf. Sci. (AiCIS),* 2018, pp. 176–181, doi: 10.1109/AiCIS.2018.00042.

[7] L. Han, F. Yuan, and Y. Jiang, "AES algorithm applied on security protocol of RFID," in *2015 3rd AASRI Conf. Comput. Intell. Bioinform. (CIB 2015),* 2015, doi: 10.12783/dtcse/cib2015/16150.

[8] O.C. Abikoye *et al.,* "Modified advanced encryption standard algorithm for information security," *Symmetry,* vol. 11, no. 12, pp. 1–16, Dec. 2019, doi: 10.3390/sym11121484.

[9] K. Eledlebi, C.Y. Yeun, E. Damiani, and Y. Al-Hammadi, "Empirical studies of TESLA protocol: Properties, implementations, and replacement of public cryptography using biometric authentication," *IEEE Access,* vol. 10, pp. 21941–21954, Feb. 2022, doi: 10.1109/access.2022.3152895.

[10] R.M.A. Al_Azzawi and S.S.M. Al-Dabbagh, "Software implementation solutions of a lightweight block cipher to secure restricted IoT environment: A review," *Al-Rafidain J. Comput. Sci. Math.*, vol. 16, no. 2, pp. 77–88, Dec. 2022, doi: 10.33899/csmj.2022.176594.

[11] M. Qasaimeh, R.S. Al-Qassas, and M. Ababneh, "Software design and experimental evaluation of a reduced AES for IoT applications," *Future Internet,* vol. 13, no. 11, pp. 1–21, Nov. 2021, doi: 10.3390/fi13110273.

[12] C. Fathy and H.M. Ali, "A secure IoT-based irrigation system for precision agriculture using the expeditious cipher," *Sensors,* vol. 23, no. 4, pp. 1–16, Feb. 2023, doi: 10.3390/s23042091.

[13] V.A. Thakor, M.A. Razzaque, and M.R.A. Khandaker, "Lightweight cryptography algorithms for resource-constrained IoT devices: A review, comparison and research opportunities," *IEEE Access,* vol. 9, pp. 28177–28193, Jan. 2021, doi: 10.1109/ACCESS.2021.3052867.

[14] G. Mudra, H. Cui, and M.N. Johnstone, "Survey: An overview of lightweight RFID authentication protocols suitable for the maritime Internet of things," *Electronics,* vol. 12, no. 13, pp. 1–20, Jul. 2023, doi: 10.3390/electronics12132990.

[15] A. Sharma and A. Singh, "Hybrid improved technique for data security and authentication for RFID tags," in *2017 4th Int. Conf. Signal Process. Comput. Control (ISPCC),* 2017, pp. 536–540, doi: 10.1109/ISPCC.2017.8269737.

[16] S. Achmad, R. Adinugroho, N.S. Hendrawan, and T. Franklin, "IoT based vehicle safety controller using Arduino," *Eng. Math. Comput. Sci. J. (EMACS),* vol. 5, no. 1, pp. 1–6, Jan. 2023, doi: 10.21512/emacsjournal.v5i1.9251.

[17] R.K. Harahap *et al.,* "Dogs feed smart system with food scales indicator IoT based," in *2022 4th Int. Conf. Cybern. Intell. Syst. (ICORIS),* 2022, pp. 1–7, doi: 10.1109/ICORIS56080.2022.10031344.

[18] S.S. Dhanda, B. Singh, and P. Jindal, "Lightweight cryptography: A solution to secure IoT," *Wireless Pers. Commun.,* vol. 112, no. 3, pp. 1947–1980, Jun. 2020, doi: 10.1007/s11277-020-07134-3.

[19] R.H. Prayitno, S.A. Sudiro, and S. Madenda, "Avoiding lookup table in AES algorithm," in *2021 6th Int. Conf. Infor. Comput. (ICIC),* 2021, pp. 1–6, doi: 10.1109/ICIC54025.2021.9632897.

[20] M.F. Moghadam *et al.,* "An efficient authentication and key agreement scheme based on ECDH for wireless sensor network," *IEEE Access,* vol. 8, pp. 73182–73192, Apr. 2020, doi: 10.1109/ACCESS.2020.2987764.

[21] B.J. Mohd and T. Hayajneh, "Lightweight block ciphers for IoT: Energy optimization and survivability techniques," *IEEE Access,* vol. 6, pp. 35966–35978, Jun. 2018, doi: 10.1109/ACCESS.2018.2848586.

[22] K.A. McKay, L. Bassham, M.S. Turan, and N. Mouha, "Report on lightweight cryptography," Nat. Inst. Stand. Technol., Gaithersburg, MD, USA, NISTIR 8114, Mar. 2017.

[23] M.Sh. Oudah and A.T. Maolood, "IoT-key agreement protocol based on the lowest work-load versions of the elliptic curve Diffie-Hellman," *Iraqi J. Sci.,* vol. 64, no. 8, pp. 4198–4207, Aug. 2023, doi: 10.24996/ijs.2023.64.8.39.

[24] M. Rashid *et al.,* "Throughput/area optimized architecture for elliptic-curve Diffie-Hellman protocol," *Appl. Sci.,* vol. 12, no. 8, pp. 1–18, Apr. 2022, doi: 10.3390/app12084091.

[25] S.Z. Khan, S.S. Jamal, A. Sajid, and M. Rashid, "FPGA implementation of elliptic-curve Diffie Hellman protocol," *Comput. Mater. Continua,* vol. 73, no. 1, pp. 1879–1894, May 2022, doi: 10.32604/cmc.2022.028152.

[26] S. Aikins-Bekoe and J.B. Hayfron-Acquah, "Elliptic curve Diffie-Hellman (ECDH) analogy for secured wireless sensor networks," *Int. J. Comput. Appl.,* vol. 176, pp. 1–8, Apr. 2020, doi: 10.5120/ijca2020920015.

[27] A.V.T. Maulidzart *et al.,* "The hybrid cryptographic algorithms for secure RFID data protection in the Internet of things," *J. ELTIKOM, J. Tek. Elekt.*

Teknol. Inf. Komput., vol. 7, no. 2, pp. 160–169, Dec. 2023, doi: 10.31961/eltikom.v7i2.860.

[28] R.H. Prayitno, S.A. Sudiro, S. Madenda, and S. Harmanto, "Hardware implementation of Galois field multiplication for mixcolumn and inversemixcolumn process in encryption-decryption algorithms," *J. Theor. Appl. Inf. Technol.*, vol. 100, no. 14, pp. 5358–5367, Jul. 2022.

[29] L. Widyawati, Husain, M. Azwar, and M.C.S. Girsang, "Analisa perbandingan hybrid cryptography RSA-AES dan ECDH-AES untuk keamanan pesan," *JUTIK (J. Teknol. Inf. Komput.)*, vol. 9, no. 2, pp. 51–62, Jan. 2023, doi: 10.36002/jutik.v9i2.2182.

[30] G. Kanda, A.O.A. Antwi, and K. Ryoo, "Hardware architecture design of AES cryptosystem with 163-bit elliptic curve," in *Adv. Multimed. Ubiquitous Eng.*, J. Park, V. Loia, K.K. Choo, and G. Yi, Eds., Singapore, Singapore: Springer, 2019, pp. 423–429, doi: 10.1007/978-981-13-1328-8_55.