Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

Design and Evaluation of a Hybrid AES-ECC Model for Secure Server Communication using REST API

Made Wisnu Adhi Saputra*[1], Roy Rudolf Huizen[2], Dandy Pramana Hostiadi[3]

[1]Magister Program, Department of Magister Information System, Institut Teknologi dan Bisnis STIKOM Bali, Indonesia
[2,3]Department of Magister Information System, Institut Teknologi dan Bisnis STIKOM Bali, Indonesia

Email: [1]232012017@stikom-bali.ac.id

Received : Jun 30, 2025; Revised : Aug 9, 2025; Accepted : Aug 9, 2025; Published : Aug 25, 2025

Abstract

Security in server-to-server communication is essential, especially in open networks vulnerable to data breaches and service disruptions. However, many existing solutions rely on a single cryptographic algorithm, limiting their ability to address diverse threats. This study aims to develop and evaluate a hybrid security model by combining the Advanced Encryption Standard (AES) and Elliptic Curve Cryptography (ECC) to ensure confidentiality, integrity, and authenticity of transmitted data. An experimental approach is applied through direct implementation in server communication. The model uses AES for symmetric encryption, ECC for dynamic session key exchange, and JSON Web Token (JWT) reinforced by nonce, timestamp, and HMAC-SHA256 for authentication and integrity verification. Test results show the model detects payload modification, replay attacks, JWT manipulation, and passive interception, with processing time still within an acceptable range. Communication efficiency is maintained with negligible payload overhead. The novelty of this research lies in integrating hybrid encryption with stateless authentication and integrity validation into a unified architecture. This integration allows security elements to be delivered systematically via REST API, making the model easy to adopt in existing architectures. The results of this study contribute to the advancement of secure API-based communication frameworks in the field of informatics, providing a practical, adaptable, and scalable solution for protecting data in distributed information systems.

Keywords : *Advanced Encryption Standard, Cryptography, Data Security, Elliptic Curve Cryptography, JSON Web Token.*

1. INTRODUCTION

Data security has become a crucial issue in the digital era, especially in communication between servers that involve continuous exchange of sensitive information. Threats such as eavesdropping, data modification, and misuse of access can occur at any time, especially in systems in open networks. The security of data stored in cloud databases is a challenging and complex issue due to malicious attacks, data breaches, and unsafe access points [1]. Therefore, a security mechanism is needed that can guarantee the confidentiality, integrity, and authenticity of data in every communication session. A robust security system must be able to detect interference and protect data from unauthorized parties. Cryptography is the main approach used to meet this need effectively. Cryptography is the practice of securing information by using keys to ensure the confidentiality, integrity, and authentication of data so that it can only be accessed by authorized parties and is protected from modification during the transmission process [2]–[4].

Advanced Encryption Standard (AES) is a symmetric block cryptography algorithm that uses the same key for encryption and decryption and is known for its speed and efficiency in securing extensive data [5]–[7]. AES was adopted as a standard by NIST in 2001 to replace DES, which had been

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

considered vulnerable to attacks, with the ability to encrypt and decrypt data efficiently [8]. AES processes data in 128-bit blocks and supports key lengths of 128, 192, or 256 bits for encryption and decryption [9]–[12]. The AES algorithm was chosen as a standard because of its high performance and speed, which has been proven to be superior to other algorithms in various encryption analyses [13], [14]. Despite its superior performance, key distribution in AES requires special attention because the key must be kept secret. If the key is intercepted during transmission, all encrypted information can be exposed.

To overcome this problem, support from asymmetric cryptographic algorithms such as Elliptic Curve Cryptography (ECC) is needed, as it enables secure key exchange. ECC is widely recognized in modern cryptography for providing high levels of security with relatively small key sizes, making it ideal for devices with limited computing resources [15], [16]. Compared to RSA, ECC offers equivalent security strength with significantly shorter keys, resulting in reduced power consumption and computational overhead [17]–[21]. In addition, ECC has advantages in managing authentication schemes, key management, low-overhead transmission efficiency, and supports privacy as well as fast verification [22]. The security of ECC relies on the mathematical hardness of the Elliptic Curve Discrete Logarithm Problem (ECDLP), for which no efficient solution exists to date using practical cryptographic parameters. Therefore, ECC remains a secure and reliable choice for asymmetric encryption [23]. Building upon this, hybrid cryptography combines the strengths of both asymmetric and symmetric algorithms, using asymmetric encryption, such as ECC, for secure key management and symmetric encryption, such as AES, for fast data encryption. This hybrid approach enhances overall system performance and security [24]. The integration of AES and ECC creates a hybrid model that balances speed and robust protection, supporting key processes such as secure key generation, exchange, derivation, and message authentication [25].

The AES-ECC hybrid model leverages the power of symmetric and asymmetric algorithms in a single system. AES is used to efficiently encrypt message content, while ECC is used to securely generate a shared secret through the Elliptic Curve Diffie-Hellman (ECDH) mechanism [26], [27]. This shared secret is used as the basis for the formation of AES keys that are not directly transmitted over the network. Through this approach, the risk of key interception can be significantly reduced. The structure of this hybrid model is also compatible with REST API-based communication systems that are commonly used in microservices and cloud computing architectures. Previous studies on hybrid cryptography have generally focused on either key exchange optimization or payload encryption performance in isolation, without integrating additional mechanisms for authentication, replay attack prevention, and integrity verification in a single unified architecture.

As a complement to the designed security model scheme, this model also integrates JSON Web Token (JWT) as a stateless authentication mechanism. JWT is a lightweight identity authentication protocol based on JSON format, which is widely used in web services because of its efficiency in transmission and processing and its essential role in ensuring data security [28]. JWT consists of three main parts, namely header (algorithm information and token type), payload (user data), and signature (the hash result of the previous element combination) [29]. Being stateless and secure, JWT is a strong candidate for authorization and session management, particularly in cloud computing environments where scalability and efficiency are paramount [30]. In this model, the JWT payload contains essential information such as a timestamp, nonce, and other metadata that serve to verify the authenticity and integrity of each request. JWT plays a crucial role in controlling time validity, preventing replay attacks, and ensuring that each transaction originates from a legitimate source. The inclusion of a nonce, a unique value generated for a single use, is very effective in combating replay attacks by ensuring that each request is unique and processed only once [31]. The use of JWT is especially relevant in distributed

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

systems that do not store user sessions directly, as it provides an efficient stateless validation mechanism without compromising security.

The novelty of this study lies in the comprehensive integration of AES-ECC hybrid cryptography with JWT-based stateless authentication, nonce and timestamp for replay attack prevention, and HMAC-SHA256 for message integrity verification into a single model. This combination not only addresses confidentiality and key exchange security but also ensures robust authentication and payload integrity in REST API-based server-to-server communication.

Therefore, this study aims to develop and test a hybrid data security model based on AES-ECC reinforced with JWT authentication for server-to-server communication. Evaluations are carried out on the encryption-decryption process, resource efficiency, payload validity, and resistance to common attacks such as replay attacks and data modification. The evaluation was conducted using a simulation of two servers and the REST API protocol in an open network. The results are expected to demonstrate that the proposed model is not only theoretically secure but also robust and efficient in real scenarios, making it a practical solution for securing data in modern, open, and stateless system architectures.

2. METHOD

This study uses a quantitative experimental approach to design and evaluate a data security model based on a combination of AES and ECC algorithms in inter-server communication. The main objective is to develop a hybrid model that guarantees data confidentiality, message integrity, and source authentication without transmitting plaintext directly. The model is designed to be implemented in a REST API architecture on an open network, taking into account process efficiency, integration flexibility, and protection against common cyber threats. The testing method is carried out in stages, including simulation of two-server communication, validation of the cryptographic process, and testing of system performance and resilience. This approach allows for a comprehensive evaluation of the effectiveness and efficiency of the proposed model.

2.1. Security Model Development

This communication model is designed for two server entities, namely Server A (sender) and Server B (receiver), which are connected via the HTTP/HTTPS protocol using the REST API endpoint. The process begins with the generation of a temporary (ephemeral) ECC key pair on the sender side, which is then used to generate a shared secret via the ECDH mechanism by utilizing the recipient's public key. From the shared secret value, a 128-bit AES session key is derived, which is then used to encrypt data using the AES algorithm in CBC mode. In each session, the model also generates a random initialization vector (IV) to increase encryption security. Data in plaintext form is then encrypted into ciphertext. To ensure the uniqueness of each request and prevent replay attacks, the model inserts a nonce and timestamp value. The Hash-based Message Authentication Code (HMAC) algorithm is used as a solution to protect data from various attacks that occur due to the nature of distributed networks [32]. In addition, HMAC is calculated based on all cryptographic elements to ensure message integrity and authentication, making it an essential component in a secure communication protocol [33].

The payload sent in the HTTP body is packaged in JSON format and consists of four elements, namely ciphertext, IV, ephemeral public key (ep_pubkey), and HMAC. As an authentication layer, a JSON Web Token (JWT) is generated separately, which contains metadata such as issuer (iss), creation time (iat), expiration time (exp), and nonce value (nonce). The JWT is then signed using the HS256 algorithm and inserted into the HTTP header with the Authorization Bearer scheme. This communication structure ensures that the encryption and authentication processes run in parallel, facilitating independent verification by the recipient. The general architecture of data communication using the AES-ECC hybrid model is visualized in Figure 1.

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

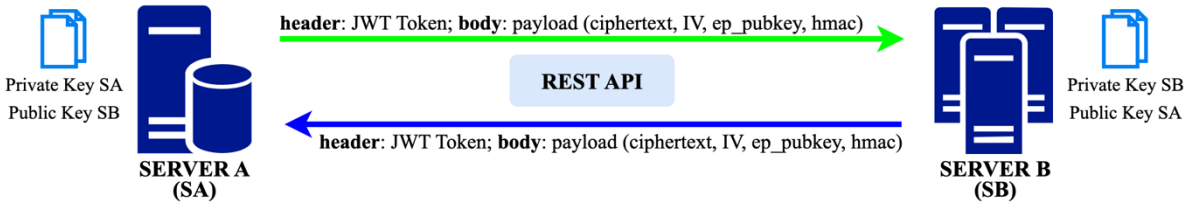

Figure 1. Secure data transmission architecture between servers using the AES-ECC hybrid model

After the payload is sent to Server B, the validation process begins by checking the header to ensure the existence and validity of the JWT. If the token is available and valid, the model checks the nonce to prevent replay attacks. The next step is to read the ephemeral public key from the sender and recalculate the shared secret using Server B's private key. Then, the session key is derived from the shared secret to be used to decrypt the ciphertext. HMAC validation is done by recalculating the value based on the payload and session key and then matching it with the received HMAC. If all verification stages are successful, the original plaintext is successfully recovered intact.

2.2. Hybrid AES-ECC Model Design

The model design is designed to support encrypted communication on the REST API architecture without compromising compatibility with existing systems. On the sender side, the process begins with the generation of an ECC ephemeral key pair, followed by the calculation of a shared secret and the derivation of a 128-bit AES session key. The plaintext is then encrypted, and validation elements such as IV, timestamp, and nonce are generated to strengthen the security against replay attacks. HMAC is calculated from all payload elements using SHA-256 to ensure integrity. A JWT token containing identity information, expiration time, and nonce is embedded in the header, and the payload is sent through the body of the REST API request. The detailed steps of the encryption and decryption process in this hybrid model are fully illustrated in Figure 2.

The diagram in Figure 2 explains the two main sides of the communication, namely the sender (Server A) and the receiver (Server B). On the sender side, the process starts by generating a temporary ECC key pair, followed by computing a shared secret, deriving an AES key, and encrypting the plaintext. After that, the system generates validation elements such as IV, timestamp, nonce, and HMAC, then forms a JSON payload and inserts a JWT token in the header.

On the recipient side, the process begins with validation of the header structure, JWT token, and nonce. If all initial verifications are successful, the model will re-form the shared secret using the sender's ephemeral public key and its private key. The AES key is re-derived and then used to decrypt the ciphertext. Message integrity is checked through HMAC recalculation, and the final result is a plaintext that is identical to the original message from the sender. This process creates a communication system that is completely encrypted and protected from manipulation and message repetition attacks.

2.3. Model Testing

Testing was conducted to assess the effectiveness of the model in a realistic server-to-server communication scenario covering aspects of functionality, efficiency, and resilience to threats. The test environment includes two servers, namely Server A, running locally with an Intel Core i5 processor and 4 GB of RAM, while Server B is running online using an Intel Xeon processor and 6 GB of RAM. The model implementation was built using the PHP 8.2 programming language and the OpenSSL library, and Postman was used for testing the REST API endpoint. Testing was conducted on various data sizes, namely 1 KB, 10 KB, 100 KB, and 1 MB, to evaluate the stability and scalability of the system in handling different loads.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

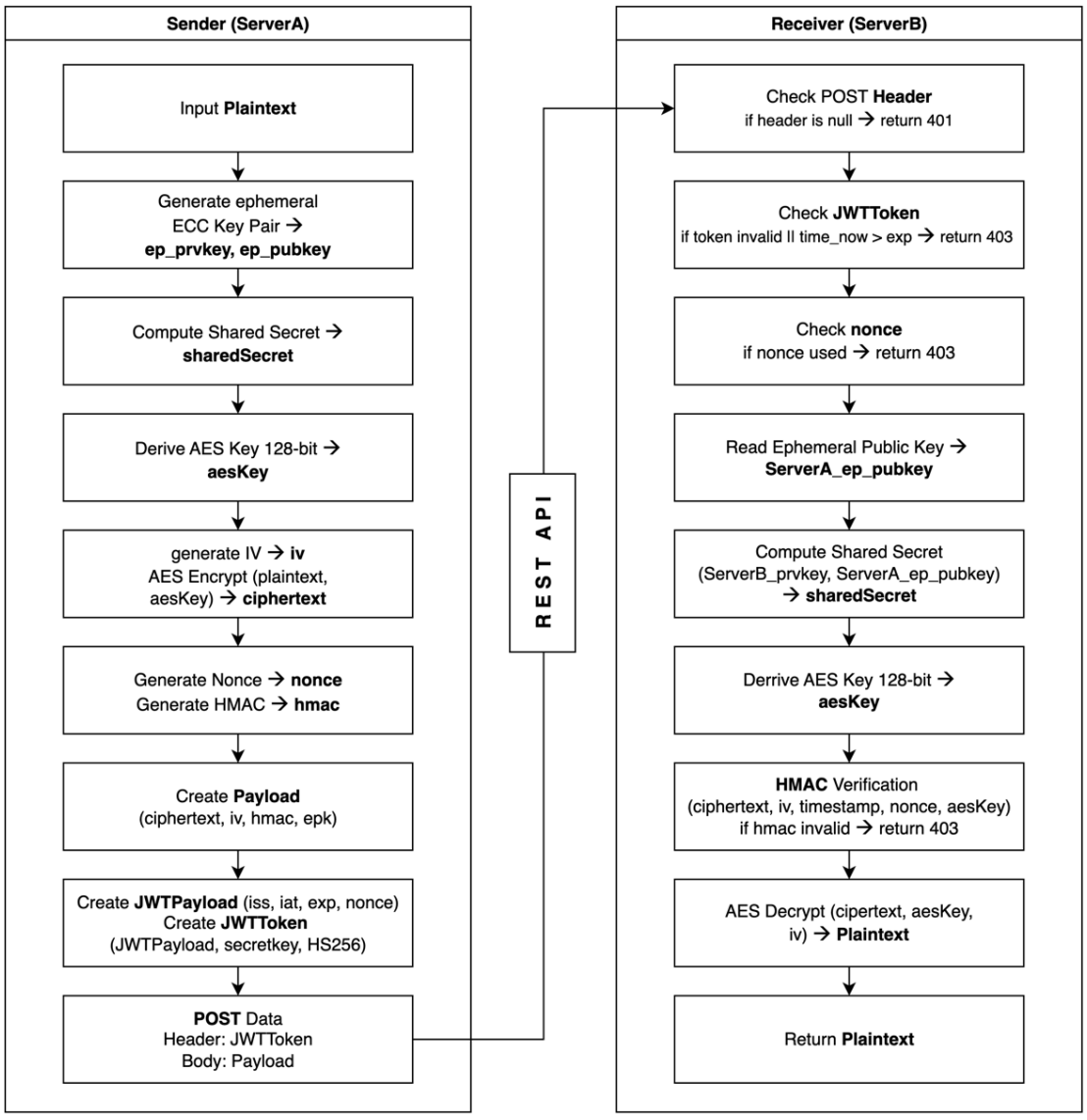

Figure 2. Data encryption and decryption process flow in the AES-ECC hybrid model

The test suite is designed to cover validation of the encryption and decryption process, measurement of processing time and memory consumption, and calculation of the overhead ratio between the payload and the original data. The test also covers various JWT-based authentication scenarios, such as empty tokens, expired tokens, and modified tokens, to assess the accuracy of the authentication mechanism. In addition, simulations of replay attacks and payload manipulation are carried out to test the effectiveness of the validation layer embedded in the system. All tests are carried out repeatedly to produce consistent data that can be compared between scenarios. Each result obtained will later be analyzed quantitatively to measure the performance and resilience of the model as a whole.

Process time measurement is done by recording timestamps before and after the cryptographic function is executed, while memory consumption is monitored using the memory_get_usage() function in each process. In addition, an evaluation is carried out on the proportion of payload size to plaintext as an indicator of data efficiency. Authentication validation is done by checking for invalid requests using invalid tokens or the same replay nonce. This test also includes verification of payload integrity by matching the HMAC value recalculated by the receiver with that sent by the sender.

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

3. RESULT

This section describes the implementation and testing results of a hybrid AES-ECC model designed to secure server-to-server communication via REST APIs. Evaluations are performed on encryption and decryption performance, memory usage, payload efficiency, JWT authentication effectiveness and system resilience to common attacks. All tests aim to assess the model in the context of an open system that represents real operational conditions.

3.1. Model Development Results

Table 1. Example of AES-ECC hybrid model encryption results

Component	Example Value
Plaintext	Design and Evaluation of a Hybrid AES-ECC Model for Secure Server Communication
IV (Initialization Vector)	"iv": "lHrFlgxxx6WyT1sCiS7Q4A=="
Ciphertext	"ciphertext": "GTI/eIe6r9KXOdOfP/xNZGRbd0Ublq+WnXhWBfYaNctuxj1nmFOla7daHyJYM5QI+rj08M5J+BIiftdFSmBuOz4NO7GgPaMuvvnPD/4yF7E=",
Ephemeral Public Key	"epk": "LS0tLS1CRUdJTiBQVUJMSUMgS0VZLS0tLS0KTUZrd0V3WUhLb1pJemowQ0FRWUlLb1pJemowREFRY0RRZ0FFVUpKcXV5OTBXNHkweTJpdWlWQTEya0FkdENPSAplKzl0U2FsN3p0azh2aWpBeUh1MGNLUE5IZzJlVE5QNHJhd055T1B3OEh3TVRncTVqQ1pGQlNUbmRRPT0KLS0tLS1FTkQgUFVCTElDIEtFWS0tLS0tCg=="
HMAC	"hmac": "IUQQ8iQenPh2rSdLkVek+ry/oFjdlYjC742+1B7EwVw="
JWT Token	Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJTZXJ2ZXIxIiwiaWF0IjoiMTc0ODYyNjEzMSIsImV4cCI6MTc0ODYyNjE5MSwibm9uY2UiOiJiNzU5YWVhOTY1NzJkMWU0ZmFmYTc1ZWY2NGY2OTQ2ZCJ9.I9jDBYpLDH2jQ4cbeLin4wgjhgCjYR—tyOktXplNLo

The AES-ECC hybrid data security model has been successfully developed to secure communication between servers via the REST API protocol. The resulting encrypted payload consists of essential cryptographic elements such as AES-encrypted ciphertext, an initialization vector (IV), an ephemeral public key (pk), and a JWT token containing iat, exp, and nonce. This token is used as a stateless authentication mechanism as well as to prevent replay attacks and limit the validity time of requests. The data structure is arranged in a standardized format so that it can be easily parsed and processed by cross-platform systems and programming languages.

On the sender side (Server A), plaintext data is encrypted using AES-128-CBC with a key derived from the shared secret of the Elliptic Curve Diffie-Hellman (ECDH) key exchange. This exchange process involves the ephemeral private key of the sender and the public key of the recipient so that the symmetric key does not need to be sent over the network. After the encryption process is complete, the model forms a JWT token that includes a timestamp (iat), expiration time limit (exp), and nonce and is then signed using the HS256 algorithm. This token is inserted into the header as authentication, while HMAC is calculated from all components of the payload to ensure data integrity.

While on the receiving side (Server B), the server validates the JWT token, including its validity period and uniqueness of the nonce to prevent replay attacks. After successful validation, the server re-forms the shared secret using the sender's ephemeral public key and its private key, then decrypts the

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

ciphertext using the derived AES key. Modified payloads, invalid JWTs, or previously used nonces will be immediately rejected. This scheme ensures that only valid requests can be processed. The representation of the model payload results can be seen in Table 1.

3.2. Performance Test Results

Performance testing was conducted to evaluate the efficiency of the AES-ECC hybrid model in a server-to-server communication scenario. The two main parameters analyzed were processing time and memory consumption during the encryption and decryption processes. The data size variations used were 1 KB, 10 KB, 100 KB, and 1 MB, representing light to large data loads. To provide a more comprehensive overview, the test results are presented in ascending order of data size, with a comparison between the encryption and decryption processes. Each measurement was performed 50 times to obtain a representative average and minimize the influence of anomalies. The encryption process includes AES key formation, IV generation, plaintext encryption, HMAC formation, and JSON payload compilation. The decryption process includes header validation, JWT validation, nonce validation, key re-formation, HMAC verification, and ciphertext decryption. The average processing time results are shown in Table 2.

Table 2. Average encryption and decryption times by data size

Data Size	Encryption Time (ms)	Decryption Time (ms)
1 KB	1.45	0.66
10 KB	1.51	0.45
100 KB	2.91	2.09
1 MB	15.95	19.15

Table 2 shows that encryption and decryption times increase with increasing data size. Encryption of 1 KB data takes 1.45 ms, while decryption only takes 0.66 ms. For 1 MB of data, the encryption time increases to 15.95 ms, and the decryption time reaches 19.15 ms. This data provides an early indication that the processing load increases linearly at small data sizes and increases sharply at extensive data. The visualization of processing time based on data size is shown in Figure 3.

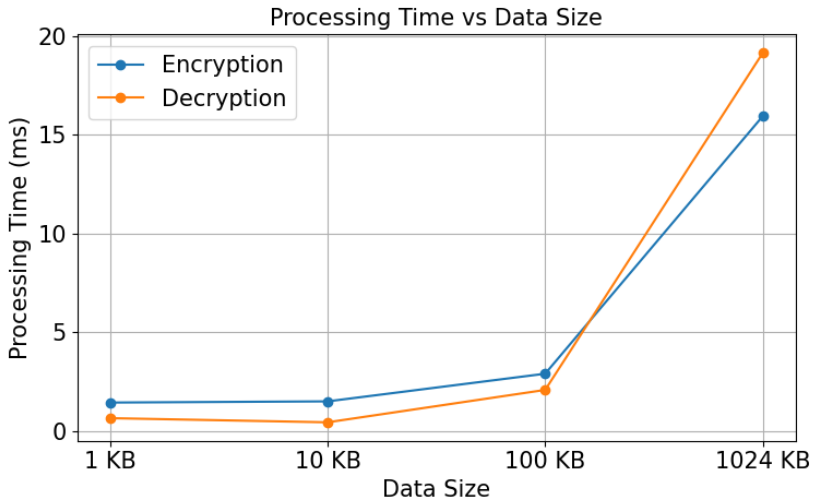

Figure 3. Encryption and decryption process time graph against data size

Figure 3 shows an exponential increase pattern when the data size reaches 1 MB, both in the encryption and decryption processes. The difference between encryption and decryption time is not too significant for small data sizes but becomes more striking for large data sizes. This shows that the

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

cryptographic processing load increases sharply for large-scale data. This information is essential to consider the efficiency of model implementation in production systems.

In addition to processing time, memory consumption is also tested to see the efficiency of system resource usage. This test is essential, especially in implementing models on servers with limited capacity. Measurements are made using the memory_get_usage() function to record memory usage during the cryptography process. The measurement results show that the encryption process tends to require more memory than decryption for the same data size. The average memory consumption is shown in Table 3.

Table 3. Memory usage during encryption and decryption

Data Size	Encryption (kb)	Description (kb)
1 KB	2.66	1.85
10 KB	16.91	12.6
100 KB	156.91	116.6
1 MB	1560.91	1172.6

To clarify the trend of increasing memory consumption based on data size, a graphical visualization is presented in Figure 4. Figure 4 shows that both encryption and decryption experience a significant spike in 1 MB data. However, the encryption process still shows a consistently larger memory usage than decryption. This visualization is essential to consider the feasibility of the model in a system with limited memory.

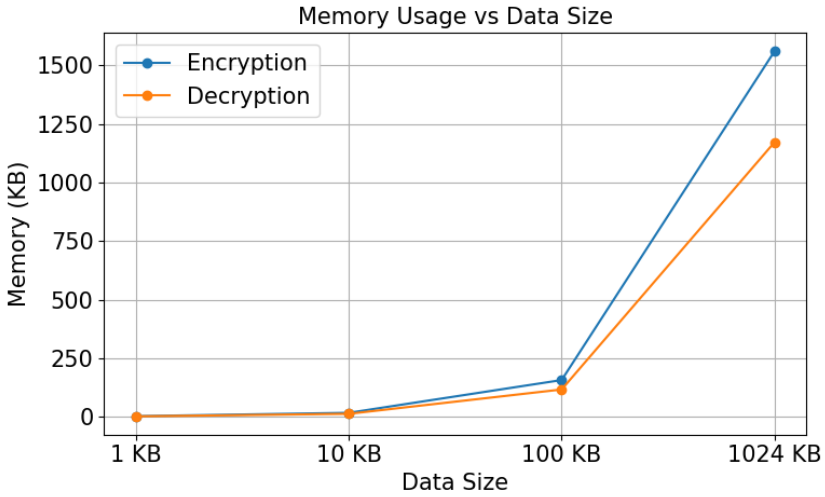

Figure 4. Graph of memory usage against data size

3.3. Data Size Test Results

Data size testing is performed to pump up the additional data size (overhead) in the payload due to the encryption process and the insertion of security elements in the AES-ECC hybrid model. The focus of this test is to compare the size of the original plaintext with the size of the encrypted payload sent over the network. Additional elements included in the payload include the initialization vector (IV), temporary public key (ep_pubkey), and HMAC. Four variations of data size were tested, namely 1 KB, 10 KB, 100 KB, and 1 MB, each tested 50 times to obtain a representative average value. All tests were conducted in a REST API network environment between two servers, with direct measurements of the encrypted payload sent. Complete test results can be seen in Table 4.

The percentage graph of overhead against data size is visualized in Figure 5. This graph shows that the model is more efficient for large data communications.

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

Table 4. Average results of payload size and percentage of overhead

Data Size	Payload Size (KB)	Overhead (%)
1 KB	1.39	20.8
10 KB	11.89	1.9
100 KB	117.30	0.2
1 MB	1197.84	0.02

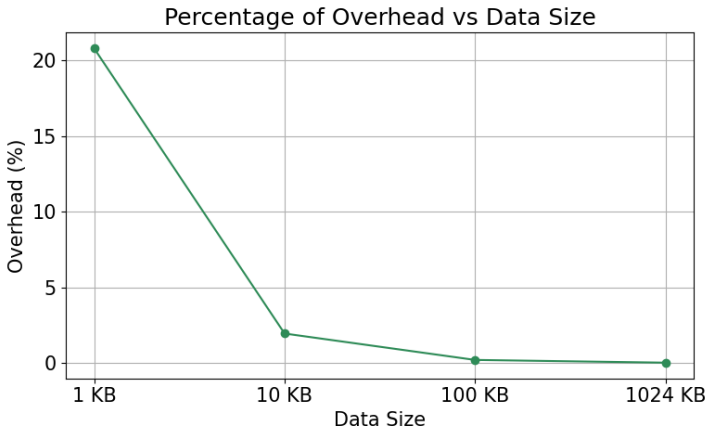

Figure 5. Percentage graph of overhead against data size

3.4. Security Test Results

Security testing was conducted to ensure that the hybrid AES-ECC model is capable of detecting and rejecting unauthorized communications, as well as protecting data from manipulation and eavesdropping. The testing covered seven common threat scenarios involving validation of HMAC, JWT, timestamp, and nonce, as well as payload analysis with Wireshark. All tests were run in a server-to-server communication environment using Postman and manual scripts.

In the normal scenario, all parameters were valid and the process ran smoothly. The payload manipulation scenario resulted in a rejection due to HMAC mismatch. Replay attacks were successfully rejected through nonce validation, while expired or unavailable JWTs were also rejected according to procedure. JWT modification was detected through failed digital signature verification. In passive interception attacks, the captured payload remained unreadable because it was fully encrypted. The test results for all scenarios are summarized in Table 5.

Table 5. Results of payload validation evaluation on various threat scenarios

No	Test Type	Scenario Description	Verification Status	Results
1	Normal	Original payload, no modified	HMAC valid, new nonce	Accepted
2	Payload modified	Ciphertext is replaced manually	Invalid HMAC	Rejected
3	Replay attack	Payload sent repeat with the same nonce	Nonce detected duplicate	Rejected
4	JWT expired	JWT sent past expiration time (exp)	Invalid timestamp	Rejected
5	JWT does not exist	JWT is not included in Request headers	JWT Not found in header	Rejected
6	JWT modified	JWT manually modified or re-signed manual	Signature JWT tidak valid	Rejected
7	MitM (passive interception/ Wireshark)	Payload captured and analyzed	Payload no readable	Safe

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

4. DISCUSSIONS

This section discusses the test results of the AES-ECC hybrid model from various aspects, including security, efficiency of inter-server communication, system performance, and its comparison with conventional encryption models. The purpose is to provide an interpretation of the research findings and assess the advantages and limitations of the proposed model. In addition, this section highlights potential challenges and limitations that may arise in specific deployment scenarios, particularly in environments with constrained computational resources.

4.1. Security Model Evaluation

The AES-ECC hybrid model demonstrates high effectiveness in maintaining three main aspects of data security, namely confidentiality, integrity, and authenticity, through the layered application of cryptographic components. Based on the test results, all threat scenarios, such as payload modification, token forgery, nonce repetition, and use of expired JWTs, can be detected and rejected consistently. This model not only secures the message content through AES encryption but also ensures that only legitimate and authentic entities can communicate through the HMAC and JWT-based validation process. The integration of these mechanisms makes the model resistant to data manipulation and replay attacks while providing process efficiency because validation is performed before the payload is further processed. This proves that the model is not only strong in Theory but also robust in practical applications in communication between servers that are vulnerable to interception and exploitation.

Security testing is carried out through simulations of various real scenarios using Postman and Wireshark, including testing ciphertext manipulation, use of the same nonce expired JWT, and absence of authentication tokens. Each test shows that the system is able to identify and block threats correctly and return appropriate error responses without further processing suspicious requests. In addition, the analysis results of the intercepted payload show that all data has been effectively encrypted so that the data cannot be read without the correct ECC private key and key derivation parameters. The success of the model in detecting and responding to threats proves that the developed model is feasible to be implemented in communication scenarios on open networks, especially on REST API infrastructures that require high-level security guarantees. However, in devices with limited processing capacity, such as IoT nodes or low-specification edge devices, the combined cryptographic operations may introduce noticeable latency, which should be considered when planning deployment in such environments.

4.2. Model Effectiveness

The effectiveness of the AES-ECC hybrid model is reflected in its success in maintaining the security of communication between servers without requiring an ongoing session. The use of JWT allows for efficient validation of time and identity, while nonce can prevent replay attacks. Based on the test results, all invalid requests were successfully rejected without a decryption process, which proves that the system works reactively and saves resources. This capability shows that the model is suitable for stateless systems such as REST APIs that prioritize efficiency and resilience in verifying each request.

This model is designed to be easily integrated into various systems that use the HTTP/HTTPS protocol, such as REST APIs and microservices-based services, without requiring additional infrastructure. Support for the JSON data format and the flexibility of its architecture allows for broad application in heterogeneous environments, both on local and cloud servers. This approach combines the advantages of AES symmetric encryption with the security of key exchange through ECC and adds a layer of authentication and integrity validation using JWT and HMAC-SHA256. Unlike the pure AES model that relies on manual key distribution, this integration enables secure and automatic key exchange. This combination makes the system more resilient to data manipulation, replay attacks, and unauthorized

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

access in communications between servers. Nonetheless, its adoption in latency-sensitive applications may require optimization strategies, such as hardware acceleration or selective use of ECC operations, to maintain responsiveness.

4.3. Performance and Efficiency Analysis

The AES-ECC hybrid model shows efficient performance in data encryption and decryption processes, with processing times that are still within a reasonable range for inter-server communication systems. Encryption times are generally higher than decryption due to additional processes such as AES key derivation and HMAC formation. Despite the additional computational load, the entire process continues to run optimally without causing significant delays, even for large data sizes. Memory usage increases as the size of the processed data increases but does not cause significant obstacles. This is due to the cryptographic mechanism that processes data gradually without loading the entire payload into memory at once. In addition, the system utilizes optimized cryptographic libraries such as OpenSSL so that memory allocation efficiency is maintained during the encryption and decryption processes. Thus, this model is suitable for use in systems with high traffic without the need for special hardware.

In terms of communication efficiency, this model inserts security elements such as Initialization Vector (IV), ephemeral public key (ep_pubkey), HMAC, and JWT into the payload to ensure data security and authenticity. This addition causes data overhead, especially for small messages such as 1 KB, where the percentage increase in size can reach more than 20%. However, for medium to extensive data, the impact of this additional element on the total payload size decreases drastically, even below 1%, so that it remains efficient in transmission. This efficiency is essential to maintain communication performance in limited networks without reducing the security layers applied. Nevertheless, the trade-off between payload overhead and security assurance needs careful consideration in bandwidth-constrained environments, as excessive overhead could still affect throughput in high-frequency transactions. Thus, the developed hybrid model provides layered protection without sacrificing overall system efficiency.

4.4. AES vs Hybrid AES-ECC Comparison

Comparisons are made for the efficiency and effectiveness of the AES-ECC hybrid model against pure AES in terms of runtime performance, payload efficiency, and security features. The test focuses on how much impact the integration of additional security components such as ECDH, HMAC, and JWT has on system performance and data overhead. With this approach, the analysis is not only limited to technical metrics but also reviews the feasibility aspects of the implementation model in a server communication environment on an open network. This comparative analysis also emphasizes the practical trade-offs between speed, security, and scalability, which are critical for decision-makers when selecting encryption approaches for different operational contexts. One of the main aspects is the encryption and decryption processing time for various data sizes, as presented in Table 6.

Table 6. Comparison of AES vs AES-ECC encryption and decryption processing time

Data Size	Encryption AES (ms)	Encryption AES-ECC (ms)	Description AES (ms)	Description AES-ECC (ms)
1 KB	0.09	1.45	0.03	0.66
10 KB	0.08	1.51	0.06	0.45
100 KB	0.35	2.91	0.19	2.09
1 MB	3.12	15.95	1.37	19.15

In general, pure AES shows faster performance because it only processes data with symmetric algorithms. Meanwhile, hybrid AES-ECC must perform additional processes such as key distribution

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

(ECDH) and HMAC calculations, which cause a significant increase in computation time. However, the resulting processing time is still in the millisecond range and is considered reasonable for a real-time system. The visualization of the comparison trend of encryption and decryption time between the two models is shown in Figure 6.

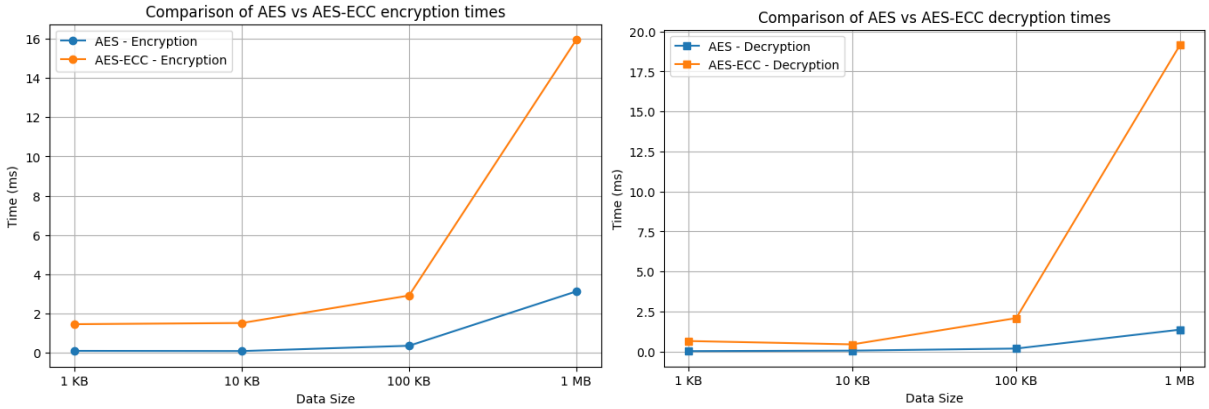

Figure 6. Graph comparison of encryption and decryption process time AES vs. AES-ECC

In terms of communication efficiency, the hybrid model adds a number of cryptographic elements to the payload, such as IV, ephemeral public key (ep_pubkey), nonce, HMAC, and JWT. This addition causes an increase in the message size, known as overhead. However, although the overhead is quite significant at small data sizes (e.g. 1 KB), this percentage decreases drastically as the data size increases. This means that the model remains efficient for large-scale communication because the overhead is no longer significant to the total message size. Details of the overhead percentages in both models are shown in Table 7.

Table 7. Comparison of payload overhead AES vs. AES-ECC

Data Size	AES (%)	AES-ECC (%)
1 KB	2.5997	20.797
10 KB	0.1458	1.946
100 KB	0.0222	0.202
1 MB	0.0023	0.020

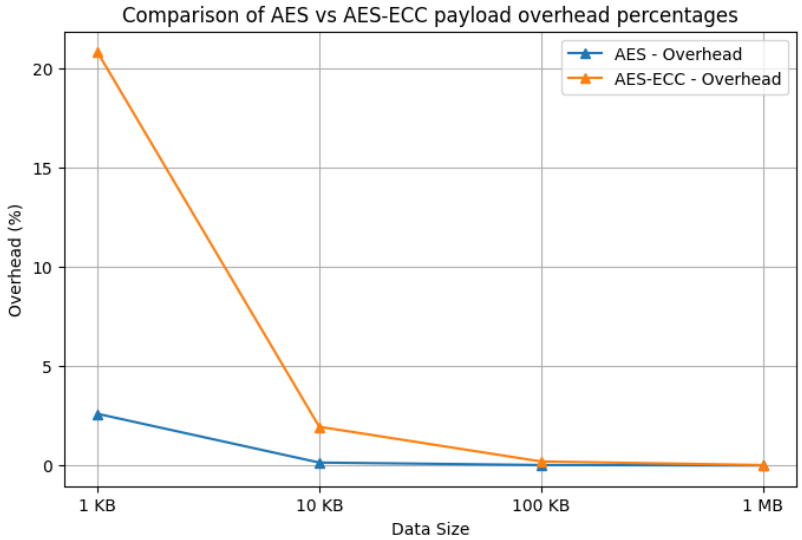

Figure 7. Graph comparison of payload overhead percentage AES vs. AES-ECC

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

The visualization of this trend can be seen in Figure 7. This graph shows that although the initial overhead is relatively high, its value approaches zero on extensive data, indicating the efficiency of the model in utilizing network bandwidth.

The following comparison is done in terms of the security features owned by each model. AES, as a symmetric algorithm, only provides data confidentiality without including additional features such as integrity validation or sender entity authentication. Even so, pure AES is still able to protect data from Man-in-the-Middle (MitM) attacks as long as the encryption key is not leaked or distributed securely. MitM attacks themselves are a type of active attack in a communications network, where an attacker infiltrates the communications path between two parties to access, intercept, or modify confidential information secretly [34]. In contrast, the hybrid AES-ECC model is able to cover all critical security aspects, from secure key distribution via ECDH and stateless authentication with JWT to replay attack mitigation and protection against data manipulation with HMAC. A summary of the comparison of these security features is presented in Table 8.

Table 8. Comparison of security features AES vs. AES-ECC

Data Size	AES	AES-ECC
Data Confidentiality	Yes	Yes
Secure Key Distribution	No	Yes
Payload Authentication	No	Yes
Replay Attack Protection	No	Yes
Validation Integrity	No	Yes
Protection MitM Attack	Limited	Yes

Overall, pure AES remains superior in speed and data size efficiency but is insufficient for advanced security needs in open network communication systems. Despite its higher overhead and more complex process, the hybrid AES-ECC model offers comprehensive protection with a performance trade-off that is still within reasonable limits. Given the increasing prevalence of sophisticated cyber threats and the critical nature of data transmitted in modern digital infrastructures, the urgency to adopt more robust encryption models such as AES-ECC becomes evident, especially in government, financial, and healthcare systems where security breaches can have severe consequences.

5. CONCLUSION

This study successfully developed and evaluated a hybrid security model that combines the Advanced Encryption Standard (AES) and Elliptic Curve Cryptography (ECC) algorithms for secure communication between servers. This model utilizes AES for data encryption, ECC for secure key distribution, and JWT containing validation elements such as timestamp, exp, and nonce to ensure message authentication and integrity. The test results show that the model is able to encrypt and decrypt data consistently, maintain payload integrity through HMAC verification, and detect various attack scenarios such as replay attacks, payload manipulation, and unauthorized JWT usage. Performance analysis also shows that the model remains efficient in terms of time and memory usage, although there is a slight overhead due to the addition of a security layer. Overall, this AES-ECC hybrid model is feasible to be applied in server-to-server communication that requires a high level of security and stateless nature. However, the scope of this research is limited to laboratory-scale simulations and controlled testing environments, without direct deployment in large-scale production systems. The evaluation did not cover extreme network conditions, very high traffic loads, or diverse hardware configurations, which may influence performance and reliability.

Further studies can be directed at the application of the AES-ECC hybrid model in the Internet of Things (IoT) ecosystem, which presents special challenges such as computational limitations, changes

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

in network topology, and the need for real-time encrypted communication. Testing on edge devices such as sensors and gateways is expected to reveal aspects of performance efficiency, model scalability, and readiness for integration in large-scale and dynamic distributed systems. From a broader perspective, the adoption of this model could have a significant impact on strengthening data protection in critical sectors such as government, healthcare, and financial services, where breaches can result in severe operational and reputational consequences.

CONFLICT OF INTEREST

The authors confirm that there are no conflicts of interest among the authors or between the authors and the subject matter of this research.

REFERENCES

[1] P. Suthanthiramani, M. Sannasy, G. Sannasi, and K. Arputharaj, "Secured Data Storage and Retrieval using Elliptic Curve Cryptography in Cloud," *International Arab Journal of Information Technology*, vol. 18, no. 1, pp. 56–66, 2021, doi: 10.34028/iajit/18/1/7.

[2] A. Hamza and B. Kumar, "A Review Paper on DES, AES, RSA Encryption Standards," *Proceedings of the 2020 9th International Conference on System Modeling and Advancement in Research Trends, SMART 2020*, pp. 333–338, 2020, doi: 10.1109/SMART50582.2020.9336800.

[3] S. Ullah, J. Zheng, N. Din, M. T. Hussain, F. Ullah, and M. Yousaf, "Elliptic Curve Cryptography; Applications, Challenges, Recent Advances, and Future Trends: A Comprehensive Survey," *Comput Sci Rev*, vol. 47, 2023, doi: 10.1016/j.cosrev.2022.100530.

[4] M. R. Khan *et al.*, "Analysis of Elliptic Curve Cryptography & RSA," *Journal of ICT Standardization*, vol. 11, no. 4, pp. 355–378, 2023, doi: 10.13052/jicts2245-800X.1142.

[5] F. Nuraeni, D. Kurniadi, and D. N. Rahayu, "Implementation of RSA and AES-128 Super Encryption on QR-Code Based Digital Signature Schemes for Document Legalization," *Jurnal Teknik Informatika (JUTIF)*, vol. 5, no. 3, pp. 675–684, 2024, doi: 10.52436/1.jutif.2024.5.3.1426.

[6] F. M. Kaffah, Y. A. Gerhana, I. M. Huda, A. Rahman, K. Manaf, and B. Subaeki, "E-Mail message Encryption using Advanced Encryption Standard (AES) and Huffman Compression Engineering," *Proceedings - 2020 6th International Conference on Wireless and Telematics, ICWT 2020*, 2020, doi: 10.1109/ICWT50448.2020.9243651.

[7] M. A. Alahe, Y. Chang, J. Kemeshi, K. Won, X. Yang, and L. Wei, "Real-Time Agricultural Image Encryption Algorithm Using AES on Edge Computing Devices," *Comput Electron Agric*, vol. 237, pp. 1–13, 2025, doi: 10.1016/j.compag.2025.110594.

[8] T. Sanida, A. Sideris, and M. Dasygenis, "Accelerating the AES Algorithm using OpenCL," *2020 9th International Conference on Modern Circuits and Systems Technologies, MOCAST 2020*, 2020, doi: 10.1109/MOCAST49295.2020.9200240.

[9] K. Muttaqin and J. Rahmadoni, "Analysis and Design of File Security System AES (Advanced Encryption Standard) Cryptography Based," *Journal of Applied Engineering and Technological Science*, vol. 1, no. 2, pp. 113–123, 2020, doi: 10.37385/jaets.v1i2.78.

[10] R. Marqas, S. M. Almufti, and R. Rebar, "Comparing Symmetric and Asymmetric Cryptography in Message Encryption and Decryption by using AES and RSA Algorithms," *Journal of Xi'an University of Architecture & Technology*, vol. 12, no. 3, 2020, doi: 10.37896/jxat12.03/262.

[11] J. Sunil, H. S. Suhas, B. K. Sumanth, and S. Santhameena, "Implementation of AES Algorithm on FPGA and on Software," *2020 IEEE International Conference for Innovation in Technology, INOCON 2020*, pp. 10–13, 2020, doi: 10.1109/INOCON50539.2020.9298347.

[12] A. Nitaj, W. Susilo, and J. Tonien, "Enhanced S-boxes for the Advanced Encryption Standard with Maximal Periodicity and Better Avalanche Property," *Comput Stand Interfaces*, vol. 87, pp. 1–5, 2024, doi: 10.1016/j.csi.2023.103769.

[13] M. S. Arman, T. Rehnuma, and M. M. Rahman, "Design and Implementation of a Modified AES Cryptography with Fast Key Generation Technique," *Proceedings of 2020 IEEE International*

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

Women in Engineering (WIE) Conference on Electrical and Computer Engineering, WIECON-ECE 2020, pp. 191–195, 2020, doi: 10.1109/WIECON-ECE52138.2020.9397992.

[14] H. J. Ali, T. M. Jawad, and H. Zuhair, "Data Security using Random Dynamic Salting and AES Based on Master-Slave Keys for Iraqi DAM Management System," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 23, no. 2, pp. 1018–1029, 2021, doi: 10.11591/ijeecs.v23.i2.pp1018-1029.

[15] A. N. Salim, T. Sutabri, E. S. Negara, and M. I. Herdiansyah, "Communication Security in the MQTT Protocol for Monitoring Internet of Things Devices using Methods Elliptic Curve Cryptography," *Jurnal Teknik Informatika (JUTIF)*, vol. 5, no. 2, pp. 377–387, 2024, doi: 10.52436/1.jutif.2024.5.2.1916.

[16] Y. Yan, "The Overview of Elliptic Curve Cryptography (ECC)," in *Journal of Physics: Conference Series*, Institute of Physics, 2022, pp. 1–8. doi: 10.1088/1742-6596/2386/1/012019.

[17] N. Josias Gbètoho Saho and E. C. Ezin, "Comparative Study on the Performance of Elliptic Curve Cryptography Algorithms with Cryptography through RSA Algorithm," in *CARI 2020 - Colloque Africain sur la Recherche en Informatique et en Mathématiques Apliquées*, Proceedings of CARI 2020, 2020. [Online]. Available: https://hal.science/hal-02926106v1

[18] J. R. Arunkumar, S. Velmurugan, B. Chinnaiah, G. Charulatha, M. R. Prabhu, and A. P. Chakkaravarthy, "Logistic Regression with Elliptical Curve Cryptography to Establish Secure IoT," *Computer Systems Science and Engineering*, vol. 45, no. 3, pp. 2635–2645, 2023, doi: 10.32604/csse.2023.031605.

[19] D. Sadhukhan, S. Ray, M. S. Obaidat, and M. Dasgupta, "A Secure and Privacy Preserving Lightweight Authentication Scheme for Smart-Grid Communication using Elliptic Curve Cryptography," *Journal of Systems Architecture*, vol. 114, 2021, doi: 10.1016/j.sysarc.2020.101938.

[20] C. Patel, A. K. Bashir, A. A. AlZubi, and R. Jhaveri, "EBAKE-SE: A Novel ECC-Based Authenticated Key Exchange Between Industrial IoT Devices using Secure Element," *Digital Communications and Networks*, vol. 9, no. 2, pp. 358–366, 2023, doi: 10.1016/j.dcan.2022.11.001.

[21] M. Rashid, O. S. Sonbul, M. Arif, F. A. Qureshi, S. S. Alotaibi, and M. H. Sinky, "A Flexible Architecture for Cryptographic Applications: ECC and PRESENT," *Computers, Materials and Continua*, vol. 76, no. 1, pp. 1009–1025, 2023, doi: 10.32604/cmc.2023.039901.

[22] Bhagappa, H. S. Divyashree, N. Avinash, B. N. Manjunatha, J. Vishesh, and M. Mamatha, "Enhancing Secrecy using Hybrid Elliptic Curve Cryptography and Diffie Hellman Key Exchange Approach and Young's Double Slit Experiment Optimizer Based Optimized Cross Layer in Multihop Wireless Network," *Measurement: Sensors*, vol. 31, 2024, doi: 10.1016/j.measen.2023.100967.

[23] K. Yokoyama, M. Yasuda, Y. Takahashi, and J. Kogure, "Complexity Bounds on Semaev's Naive Index Calculus Method for ECDLP," *Journal of Mathematical Cryptology*, vol. 14, no. 1, pp. 460–485, 2020, doi: 10.1515/jmc-2019-0029.

[24] B. Ranganatha Rao and B. Sujatha, "A Hybrid Elliptic Curve Cryptography (HECC) Technique for Fast Encryption of Data for Public Cloud Security," *Measurement: Sensors*, vol. 29, pp. 1–12, 2023, doi: 10.1016/j.measen.2023.100870.

[25] A. Tidrea, A. Korodi, and I. Silea, "Elliptic Curve Cryptography Considerations for Securing Automation and SCADA Systems," *Sensors*, vol. 23, no. 5, 2023, doi: 10.3390/s23052686.

[26] R. Qazi, K. N. Qureshi, F. Bashir, N. U. Islam, S. Iqbal, and A. Arshad, "Security Protocol using Elliptic Curve Cryptography Algorithm for Wireless Sensor Networks," *J Ambient Intell Humaniz Comput*, vol. 12, no. 1, pp. 547–566, 2021, doi: 10.1007/s12652-020-02020-z.

[27] S. Di Matteo, L. Baldanzi, L. Crocetti, P. Nannipieri, L. Fanucci, and S. Saponara, "Secure Elliptic Curve Crypto-Processor for Real-Time IoT Applications," *Energies (Basel)*, vol. 14, no. 15, 2021, doi: 10.3390/en14154676.

[28] L. Zhang, C. Zhou, and J. Wen, "APSH-JWT: an Authentication Protocol Based on JWT With Scalability and Heterogeneity in Edge Computing," *Wireless Networks*, vol. 31, pp. 2939–2953, 2025, doi: 10.1007/s11276-025-03926-2.

Vol. 6, No. 4, August 2025, Page. 2740-2755
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.4.4989

[29] S. Ahmad, M. Arif, J. Ahmad, and S. Mehfuz, "A TOTP-Based Secure Data Storage System in the Cloud Environment using the JWT Token Approach," *International Journal of System Assurance Engineering and Management*, vol. 16, no. 4, pp. 1565–1578, 2025, doi: 10.1007/s13198-025-02775-8.

[30] A. S. Shatnawi, B. Al-Duwairi, and A. A. Samarneh, "Comprehensive Empirical Study of Python JWT Libraries," in *Procedia Computer Science*, Elsevier B.V., 2024, pp. 827–832. doi: 10.1016/j.procs.2024.06.099.

[31] F. De Rango, G. Potrino, M. Tropea, and P. Fazio, "Energy-aware Dynamic Internet of Things Security System based on Elliptic Curve Cryptography and Message Queue Telemetry Transport Protocol for Mitigating Replay Attacks," *Pervasive Mob Comput*, vol. 61, pp. 3–17, 2020, doi: 10.1016/j.pmcj.2019.101105.

[32] K. Karthikeyan and P. Madhavan, "Building a Trust Model for Secure Data Sharing (TM-SDS) in Edge Computing Using HMAC Techniques," *Computers, Materials and Continua*, vol. 71, no. 3, pp. 4183–4197, 2022, doi: 10.32604/cmc.2022.019802.

[33] C. E. Castellon, S. Roy, O. P. Kreidl, A. Dutta, and L. Boloni, "Towards an Energy-Efficient Hash-based Message Authentication Code (HMAC)," in *2022 IEEE 13th International Green and Sustainable Computing Conference, IGSC 2022*, Institute of Electrical and Electronics Engineers Inc., 2022. doi: 10.1109/IGSC55832.2022.9969377.

[34] B. Ahamed, F. Kareem, and M. Y. Noor Mohamed, "Advancement of The ECC Algorithm to Prevent Man-in-the-Middle and Replay Attacks," in *Procedia Computer Science*, Elsevier B.V., 2025, pp. 1259–1269. doi: 10.1016/j.procs.2025.04.081.