Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 15, No. 3, 2026

Design and Implementation of a Secure Data Transmission Protocol for IoT Devices in Smart Grids

Zhanghao Wei *

School of Electronic and Communication Engineering, Sun Yat-sen University, Shenzhen, Guangdong, 518000, China
*** Corresponding author Email:** weizhh28@mail2.sysu.edu.cn

Abstract: With the widespread integration of Internet of Things (IoT) technologies into smart grid systems, a large number of intelligent terminal devices are required to transmit data through networks in real time. However, the open network environment exposes data communication to various security threats, including eavesdropping, tampering, and replay attacks. Considering the limited computational capability and storage resources of IoT devices in smart grids, this study proposes a secure data transmission protocol based on elliptic curve cryptography and a lightweight identity authentication mechanism. The proposed protocol employs elliptic curve cryptography to accomplish key exchange and integrates the AES symmetric encryption algorithm to construct a hybrid encryption scheme, thereby improving encryption efficiency. In addition, device authentication is achieved through an identity-based authentication mechanism combined with random numbers and hash functions. Message authentication codes and a retransmission mechanism are further introduced to ensure data integrity and transmission reliability. The protocol was implemented on a Python platform and experimentally evaluated on representative IoT hardware environments, including STM32F407 and ESP32 devices. Experimental results demonstrate that, compared with the conventional RSA+AES encryption scheme, the proposed protocol exhibits significant advantages in encryption efficiency, communication latency, and computational overhead, while effectively resisting eavesdropping, tampering, and replay attacks. The findings indicate that the proposed protocol ensures robust security while maintaining favorable computational efficiency and communication performance, thereby meeting the practical requirements for secure communication among IoT devices in smart grid environments.

Keywords: Smart Grid; Internet of Things; Data Transmission Protocol; Security; Elliptic Curve Cryptography.

1. Introduction

With the rapid development of the Energy Internet and information and communication technologies, the smart grid has gradually emerged as a key direction in the evolution of modern power systems. By integrating advanced sensing technologies, communication networks, and data analytics platforms, smart grids enable real-time monitoring and intelligent management of power system operations. In recent years, the application of the Internet of Things (IoT) in smart grids has expanded significantly. A large number of intelligent terminal devices—including smart meters, environmental sensors, remote terminal units, and distributed energy control devices—have been deployed within the perception and control layers of the power grid. These devices continuously collect operational data from the grid and transmit it through communication networks to data centers for analysis and decision-making, thereby improving operational efficiency and energy utilization. However, as massive numbers of IoT devices are integrated into power systems, the security of data communication has become an increasingly critical concern. If attacks such as eavesdropping, tampering, or data forgery occur during transmission, they may disrupt grid operation decisions and even lead to severe power system incidents. Therefore, designing secure and efficient data transmission mechanisms for resource-constrained IoT environments has become a key research challenge in smart grid systems [1].

As the scale of smart grids continues to expand, the security threats faced by communication networks have become increasingly complex. For example, attackers may attempt to disrupt system communications through replay attacks, man-in-the-middle attacks, or denial-of-service attacks, thereby compromising the confidentiality, integrity, and availability of grid data [2]. Conventional network security mechanisms typically require substantial computational resources and storage capacity, which are difficult for resource-limited IoT devices to support. Consequently, researchers have begun to explore lightweight secure communication mechanisms tailored to IoT environments in order to ensure system security while minimizing computational and communication overhead [3]. Among the available cryptographic techniques, elliptic curve cryptography (ECC) has become an important approach for securing IoT communications because it can provide strong security with relatively short key lengths. Compared with the traditional RSA public-key cryptosystem, ECC requires significantly shorter keys to achieve the same security level, thereby reducing computational complexity and communication overhead. This characteristic makes ECC particularly suitable for resource-constrained embedded devices and sensor nodes [4]. In recent years, a variety of lightweight authentication protocols based on ECC have been proposed to address secure communication challenges in smart grid and IoT environments [5].

Furthermore, to enhance both communication security and efficiency, several studies have begun to integrate symmetric encryption techniques with ECC, forming hybrid encryption mechanisms. In such schemes, ECC is typically employed for secure key exchange, while efficient symmetric encryption algorithms—such as AES—are used to encrypt the transmitted data. This approach achieves a balance between strong security and computational efficiency. Previous studies have shown that hybrid encryption schemes can significantly improve data transmission efficiency while maintaining a high level of security in complex network environments [6].

Meanwhile, in response to security threats in IoT environments, additional mechanisms—such as timestamps, random numbers, and message authentication codes—have been incorporated into communication protocols to enhance system resilience against replay attacks and data tampering [7]. Although existing research has made notable progress in improving communication security within smart grid systems, several limitations remain. For instance, some security protocols do not sufficiently consider the resource-constrained characteristics of IoT devices, resulting in considerable computational overhead. Other protocols offer strong security guarantees but involve high communication complexity, which limits their applicability in large-scale IoT deployments. Therefore, designing a data transmission protocol that simultaneously balances security, computational efficiency, and communication efficiency remains an important research challenge [8].

In response to these issues, this study proposes a secure data transmission protocol for IoT devices in smart grids that integrates elliptic curve cryptography with a lightweight identity authentication mechanism. The proposed protocol employs ECC to achieve key exchange and device authentication, while symmetric encryption is used to protect transmitted data, thereby improving transmission efficiency and reducing computational overhead. In addition, message authentication codes and random-number mechanisms are incorporated to ensure data integrity and prevent replay attacks. Experimental results demonstrate that the proposed protocol outperforms the traditional RSA-based encryption scheme in terms of both security and communication efficiency, making it well suited to meet the secure communication requirements of IoT devices in smart grid environments.

2. Related Technical Foundations

2.1. ECC

ECC is a public-key cryptographic algorithm based on the mathematical theory of elliptic curves. It was independently proposed in 1985 by Neal Koblitz and Victor S. Miller. Compared with the traditional RSA algorithm, ECC offers several notable advantages, including shorter key lengths, higher computational efficiency, and strong security performance. The fundamental principle of ECC is to utilize point operations on elliptic curves to perform encryption and decryption. The general form of an elliptic curve equation can be expressed as $y^2 = x^3 + ax + b$, where a and b are parameters that define the curve. In ECC systems, each user generates a pair of cryptographic keys consisting of a public key and a private key. The public key is used for encrypting data, while the private key is used for decryption. During the encryption process, the sender encrypts the data using the receiver's public key, and the receiver subsequently decrypts the ciphertext using their private key. Owing to the mathematical properties of elliptic curves, ECC can provide a level of security comparable to that of RSA while using significantly shorter keys. For instance, a 256-bit ECC key offers a security level roughly equivalent to that of a 3072-bit RSA key. As a result, ECC is particularly well suited for resource-constrained environments, such as IoT devices deployed in smart grids, including sensors and embedded controllers.

2.2. Lightweight Authentication Mechanisms

Lightweight authentication mechanisms are authentication approaches specifically designed for resource-constrained IoT devices. Their primary objective is to ensure communication security while reducing computational overhead and communication latency. Common lightweight authentication approaches include hash-based authentication, identity-based authentication, and authentication based on Physical Unclonable Function (PUF). Hash-based authentication generates authentication codes by applying hash operations to device identity information and random numbers. The receiver verifies the legitimacy of the authentication code to confirm the identity of the device. Identity-based authentication, in contrast, utilizes the user's identity information—such as a device ID—as the public key, thereby avoiding the complex management requirements associated with traditional Public Key Infrastructure (PKI). PUF-based authentication exploits the intrinsic physical characteristics of hardware devices, such as manufacturing variations in integrated circuits, to generate unique authentication information. Because these physical characteristics are difficult to replicate, PUF-based approaches provide strong security and resistance to cloning attacks. In smart grid environments, lightweight authentication mechanisms can effectively prevent device identity spoofing and replay attacks, thereby enhancing the overall security of data transmission.

2.3. IoT Architecture in Smart Grids

The IoT architecture in smart grid systems generally consists of three layers: the perception layer, the network layer, and the application layer. The perception layer is responsible for data acquisition and typically includes devices such as smart meters, sensors, and controllers. These devices collect operational data from the power system and transmit the collected information to the network layer through wireless communication technologies, such as ZigBee, LoRa, and NB-IoT. The network layer is responsible for data transmission and routing. It includes communication infrastructure such as gateways, routers, and switches, which relay the data collected by the perception layer to the application layer. The application layer focuses on data processing and service implementation, encompassing systems such as power dispatching, load management, and fault diagnosis. These systems analyze and process the collected data to support intelligent monitoring and management of the smart grid. Within the IoT architecture of smart grids, data exchange among different layers must satisfy both security and real-time requirements. Consequently, the design of efficient and secure data transmission protocols is essential to ensure reliable communication across the entire system.

3. Secure Data Transmission Protocol Design

3.1. Protocol Architecture

The proposed protocol adopts a layered design consisting of three components: the application layer, the transport layer, and the security layer. The application layer is responsible for data generation and processing, while the transport layer handles data transmission across the communication network. The security layer performs critical security functions,

including data encryption, identity authentication, and integrity verification. Data interaction between the application layer and the transport layer is implemented through application programming interfaces (APIs), enabling efficient data exchange between system modules. Meanwhile, communication between the transport layer and the security layer is conducted through a dedicated security protocol that ensures the confidentiality, authenticity, and integrity of transmitted data. This layered architecture enhances system modularity and flexibility while providing a structured framework for secure data transmission in smart grid IoT environments.

3.2. Identity Authentication Mechanism

A lightweight identity-based authentication mechanism is adopted in the proposed protocol. Devices generate authentication information using a pre-shared key combined with a cryptographic hash function. The authentication procedure is described as follows.

Device registration: When a new device joins the network, it first sends a registration request to the server. The server then assigns the device a unique device identifier (Device ID) and a pre-shared key, which are used for subsequent authentication and communication.

Identity authentication: Before transmitting data, the device initiates an authentication request to the server, including its Device ID and a randomly generated nonce. Upon receiving the request, the server retrieves the corresponding pre-shared key based on the Device ID and performs a hash operation on the random number to generate an authentication code. The authentication code is then returned to the device. After receiving the authentication code, the device performs the same hash computation on the random number using the pre-shared key and verifies the validity of the authentication code.

Session key generation: Once authentication is successfully completed, the server generates a session key for secure communication. The session key is encrypted using the device's public key and transmitted to the device. After receiving the encrypted message, the device decrypts it using its private key to obtain the session key, which is subsequently used for data encryption and decryption during secure communication.

3.3. Data Encryption and Decryption

The proposed protocol employs an encryption scheme based on Elliptic Curve Cryptography to secure transmitted data. The encryption and decryption process is described as follows.

Key generation: Each device generates a pair of cryptographic keys consisting of a public key and a private key. The public key is transmitted to the server, while the private key is securely stored on the device.

Data encryption: During data transmission, the sender encrypts the data using the receiver's public key, thereby generating ciphertext that can be securely transmitted over the network.

Data decryption: Upon receiving the ciphertext, the receiver decrypts it using its private key to recover the original plaintext data.

To further improve encryption efficiency, a hybrid encryption approach is adopted. Specifically, a symmetric encryption algorithm—such as Advanced Encryption Standard—is used to encrypt the transmitted data, while elliptic curve cryptography is employed to encrypt the symmetric key. This hybrid scheme combines the high efficiency of symmetric encryption with the strong security properties of public-key cryptography.

3.4. Integrity Verification and Retransmission Mechanism

To ensure data integrity during transmission, a message authentication code (MAC) is generated using a cryptographic hash function. The receiver verifies the MAC value to confirm that the transmitted data has not been altered during communication. The procedure is described as follows.

MAC generation: The sender performs a hash operation on the transmitted data to generate a MAC value. This MAC value is appended to the data packet and transmitted together to the receiver.

MAC verification: Upon receiving the data, the receiver performs the same hash computation on the received message to generate a new MAC value. This value is then compared with the MAC attached to the received data. If the two values are identical, the data is considered intact; otherwise, the data may have been tampered with during transmission.

To further ensure transmission reliability, a retransmission mechanism is implemented to address potential data loss or corruption during communication. The process is described as follows.

Timeout-based retransmission: After transmitting a data packet, the sender initiates a timer. If an acknowledgment from the receiver is not received before the timer expires, the sender retransmits the data packet.

Acknowledgment mechanism: Once the receiver successfully receives the data, it sends an acknowledgment message to the sender. Upon receiving this acknowledgment, the sender terminates the timer and proceeds with subsequent transmissions.

Through the combination of these mechanisms, the proposed protocol effectively guarantees the confidentiality, integrity, and availability of transmitted data, thereby meeting the security requirements of IoT devices in smart grid environments.

4. Protocol Implementation and Testing

4.1. Protocol Implementation

The protocol was implemented using the Python programming language. Cryptographic operations, including encryption and decryption, were carried out using the Python cryptography library, while network communication was implemented through the Python socket library. Specifically, elliptic curve cryptographic operations were implemented using the cryptography. hazmat. primitives. asymmetric module, whereas symmetric encryption based on the Advanced Encryption Standard was realized through the cryptography. hazmat. primitives.ciphers module. In addition, the Python hashlib library was employed to implement hash functions for generating MACs.

Two commonly used IoT development platforms were selected as the experimental testbed: the STM32F407 microcontroller development board based on the ARM Cortex-M4 core, and the ESP32 development board. The STM32F407 board provides relatively strong computational capability and a wide range of peripheral interfaces, making it suitable for protocol validation experiments. The ESP32

chip integrates Wi-Fi and Bluetooth communication modules, making it particularly appropriate for testing wireless communication scenarios. The implementation process consists of the following steps.

Key generation: An ECC-based key pair, including a private key and a public key, is generated for each device. The private key is securely stored on the device, while the public key is used for encryption. During key generation, the generate_private_key function from the cryptography.hazmat.primitives.asymmetric.ec module is used, and the secp256r1 elliptic curve is selected.

Data encryption: Data encryption is performed using the AES algorithm. The encryption key is derived from the shared key generated through the ECC key exchange process. During encryption, the Cipher class from the cryptography. hazmat. primitives. ciphers module is used, with the AES algorithm operating in GCM mode.

Data transmission: Network communication is implemented using the socket library to establish a TCP connection, through which encrypted data is transmitted to the receiver. In this process, the socket.socket function is used to create a socket, the connect function is used to establish a connection with the receiver, and the send function is used to transmit the encrypted data.

Data decryption: Upon receiving the data, the receiver decrypts it using the AES algorithm. The decryption key is derived from the shared key generated through the ECC key exchange mechanism. Similar to the encryption process, the Cipher class from the cryptography. hazmat. primitives. ciphers module is used with AES operating in GCM mode.

4.2. Testing Environment and Parameters

A simulated IoT environment for a smart grid system was constructed to evaluate the proposed protocol. The experimental setup consisted of multiple IoT devices and a centralized data center. The IoT devices were responsible for collecting operational data and transmitting it to the data center, while the data center handled data reception, processing, and analysis. To comprehensively assess the performance of the protocol, several testing parameters were configured, including data size, encryption algorithms, and network conditions. These parameters were varied across different experimental scenarios in order to evaluate the protocol's performance under diverse operating conditions. The specific configuration of the testing parameters is summarized in Table 1.

Table 1. Testing Environment and Parameters

Parameter	Value
Data size	1 KB, 10 KB, 100 KB
Encryption algorithm	ECC + AES, RSA + AES
Network environment	Local area network (LAN), wide area network (WAN), high packet-loss network
Hardware platform	STM32F407 development board, ESP32 development board, standard PC
Software environment	Python 3.8, cryptography 3.4.8, socket

4.3. Experimental Results

Under the aforementioned testing environment and parameter settings, the proposed protocol was evaluated in terms of confidentiality, integrity, and availability. The experimental results demonstrate that the protocol effectively ensures secure data transmission and can resist several common network attacks, including eavesdropping, tampering, and replay attacks. For example, in the eavesdropping attack test, network packets were captured using the Wireshark tool. The results showed that the intercepted packets were encrypted ciphertext, and the original data could not be directly interpreted. In the tampering attack test, the transmitted data were intentionally modified during transmission. Upon receiving the altered message, the receiver verified the MAC and detected the inconsistency, thereby rejecting the compromised data. In the replay attack test, the protocol employed timestamp and random-number mechanisms, which effectively prevented attackers from reusing previously captured messages.

In addition, the computational overhead and communication latency of the protocol were evaluated, and the results are summarized in Table 2. The experimental findings indicate that the proposed protocol introduces relatively low computational overhead and communication delay, enabling it to meet the real-time communication requirements of IoT devices deployed in smart grid environments. As shown in Table 2, the protocol employing the ECC+AES encryption scheme demonstrates superior performance in terms of both computational overhead and communication latency compared with the protocol based on the RSA+AES scheme. This improvement can be attributed to the higher computational efficiency and shorter key length of elliptic curve cryptography, which effectively reduces both processing costs and transmission delay.

Table 2. Results of Computational Overhead and Communication Latency

Test Metric	ECC+AES	RSA+AES
Encryption time (ms)	5.2	12.8
Decryption time (ms)	4.8	11.5
Signature generation time (ms)	3.1	8.2
Signature verification time (ms)	2.9	7.8
Average communication latency (ms)	68	120

In addition, the impact of different data sizes on protocol performance was evaluated, and the results are presented in Table 3. The results indicate that as the data size increases, both computational overhead and communication latency also increase; however, the magnitude of this increase remains relatively limited. When the data size increases from 1 KB to 100 KB, the encryption time rises from 5.2 ms to 10.2 ms, while the decryption time increases from 4.8 ms to 9.8 ms. Meanwhile, the average communication latency grows from 68 ms to 92 ms. These results suggest that the proposed protocol maintains relatively high performance even when handling larger volumes of data.

Table 3. Test Results for Different Data Sizes

Data Size	Encryption Time (ms)	Decryption Time (ms)	Average Communication Latency (ms)
1 KB	5.2	4.8	68
10 KB	6.5	6.1	75
100 KB	10.2	9.8	92

The impact of different network environments on the performance of the proposed protocol was also evaluated, and the results are summarized in Table 4. The findings indicate that the protocol maintains stable performance under various network conditions and demonstrates strong adaptability to complex communication environments. Specifically, in a LAN environment, the protocol exhibits an average communication latency of 68 ms with a packet loss rate of 0.1%. In a WAN environment, the average communication latency increases to 120 ms, while the packet loss rate reaches 1.2%. Under high packet-loss network conditions, the average communication latency rises to 250 ms, with a packet loss rate of 5.0%. Overall, these results indicate that the proposed protocol maintains reliable communication performance across different network environments and can effectively adapt to complex and unstable network conditions.

Table 4. Test Results under Different Network Environments

Network Environment	Average Communication Latency (ms)	Packet Loss Rate (%)
LAN	68	0.1
WAN	120	1.2
High packet-loss network	250	5.0

The influence of different hardware platforms on the performance of the proposed protocol was also evaluated, and the results are summarized in Table 5. The results indicate that the protocol maintains satisfactory performance across different hardware platforms and demonstrates good adaptability to diverse hardware environments. Specifically, on a standard PC platform, the protocol achieves an encryption time of 5.2 ms, a decryption time of 4.8 ms, and an average communication latency of 68 ms. On the STM32F407 development board, the encryption time is 8.5 ms, the decryption time is 7.8 ms, and the average communication latency is 85 ms. On the ESP32 development board, the encryption time increases to 12.1 ms, the decryption time is 11.5 ms, and the average communication latency reaches 105 ms. These results indicate that although computational performance varies across platforms due to differences in hardware capabilities, the proposed protocol maintains stable and acceptable performance in all tested environments, demonstrating its suitability for deployment on heterogeneous IoT hardware platforms.

Table 5. Test Results on Different Hardware Platforms

Hardware Platform	Encryption Time (ms)	Decryption Time (ms)	Average Communication Latency (ms)
STM32F407 development board	8.5	7.8	85
ESP32 development board	12.1	11.5	105
Standard PC	5.2	4.8	68

In summary, the proposed protocol ensures a high level of communication security while maintaining relatively low computational overhead and communication latency. Consequently, it is capable of meeting both the security requirements and the real-time performance demands of IoT devices deployed in smart grid environments.

5. Conclusion

This study addresses the security risks and resource constraints encountered by IoT devices during data transmission in smart grid environments and proposes a secure data transmission protocol based on elliptic curve cryptography and a lightweight authentication mechanism. The protocol integrates ECC-based public key encryption with the AES symmetric encryption algorithm to construct a hybrid encryption framework, thereby achieving both secure key exchange and efficient protection of data confidentiality. In addition, an identity-based lightweight authentication mechanism is introduced. Through the combined use of random numbers, hash functions, and session key negotiation, the protocol enables reliable device authentication and the establishment of secure communication channels. Furthermore, the incorporation of MACs and an acknowledgment–retransmission mechanism enhances the integrity and reliability of data transmission.

In terms of implementation, the protocol was developed on the Python platform and experimentally evaluated on representative IoT hardware platforms, including STM32F407 and ESP32 devices. Experimental results indicate that, compared with the conventional RSA+AES scheme, the proposed ECC+AES approach demonstrates clear advantages in encryption time, decryption time, and communication latency. Under identical testing conditions, the average encryption time and communication delay of ECC+AES were significantly lower than those of RSA+AES, indicating that the proposed protocol can effectively reduce computational overhead while improving communication efficiency. Moreover, several network security scenarios—including eavesdropping, data tampering, and replay attacks—were simulated to evaluate the robustness of the protocol. The results confirm that the proposed approach provides effective protection in terms of confidentiality, data integrity, and resistance to common network attacks.

Overall, the proposed secure data transmission protocol ensures strong security while maintaining relatively low computational complexity and communication overhead. It is therefore capable of meeting the dual requirements of security and real-time performance for IoT devices in smart grid systems. Future research may further integrate emerging

technologies such as blockchain, edge computing, and zero-trust security architectures to enhance the scalability and overall security of smart grid communication systems.

References

[1] Jebri S, Amor A B, Zidi S. Reliable low-cost data transmission in smart grid system[J]. Computer communications, 2024, 214: 174-183.

[2] Ferrag M A, Maglaras L, Moschoyiannis S, et al. Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study[J]. Journal of Information Security and Applications, 2020, 50: 102419.

[3] Khan M A, Salah K. IoT security: Review, blockchain solutions, and open challenges[J]. Future generation computer systems, 2018, 82: 395-411.

[4] Chhikara D, Rana S, Mishra A, et al. Construction of elliptic curve cryptography-based authentication protocol for internet of things[J]. Security and Privacy, 2022, 5(4): e226.

[5] Wang C, Huo P, Ma M, et al. A provable secure and lightweight ECC-based authenticated key agreement scheme for edge computing infrastructure in smart grid[J]. Computing, 2023, 105(11): 2511-2537.

[6] Xiong J, Shen L, Liu Y, et al. Enhancing IoT security in smart grids with quantum-resistant hybrid encryption[J]. Scientific Reports, 2025, 15(1): 3.

[7] Rosa P, Souto A, Cecílio J. Light-SAE: A lightweight authentication protocol for large-scale IoT environments made with constrained devices[J]. IEEE Transactions on Network and Service Management, 2023, 20(3): 2428-2441.

[8] Tanveer M, Alasmary H. LACP-SG: Lightweight authentication protocol for smart grids[J]. Sensors, 2023, 23(4): 2309.