Research on Encryption Algorithm and Embedded System Optimization Strategy Based on IoT Security

Jing Liang

Department of Network and Information Security, Chongqing Vocational Institute of Safety Technology, Wanzhou Chongqing 404020, China
E-mail: cqanquanwax@163.com

Received 30 November 2024; Accepted 27 January 2025

Abstract

In the face of increasingly complex IoT environments and increasing data volumes, existing encryption algorithms still need to be further optimized in terms of the balance between efficiency and security. For embedded systems, their hardware resources are limited, and the current optimization strategies still have shortcomings in improving system performance, reducing power consumption and enhancing system stability. In this study, the optimization strategy of encryption algorithm and embedded system based on IoT security is studied. Designed for lightweight encryption algorithms and deployed in embedded systems, it aims to balance the security and performance of IoT devices and provide users with a seamless and reliable service experience. In this study, three encryption algorithms, AES-Light, SPECK and SIMON, were selected and compared on ARM Cortex-M series microcontrollers. Experiments show that the SPECK algorithm leads with its excellent encryption and decryption rate, which is 15% faster than AES-Light, while the power consumption of SIMON is reduced by 20%. Based on this, preferred encryption schemes suitable for different IoT scenarios are established. In addition, in order to overcome the limitation of fixed encryption settings in a

Journal of Cyber Security and Mobility, Vol. 14_1, 229–258.
doi: 10.13052/jcsm2245-1439.14110

dynamic network environment, this project proposes an adaptive encryption strength adjustment strategy. By monitoring the risk level of the equipment in real-time and automatically optimizing the encryption parameters, the security guarantee is improved by 30% in high-risk situations while avoiding unnecessary computing overhead in low-risk scenarios, improving the overall efficiency by more than 15%, and significantly enhancing the intelligence and adaptability of IoT systems.

Keywords: IoT security, encryption algorithm, embedded system, performance optimization.

1 Introduction

Today, with the rapid development of information technology, the Internet of Things (IoT) is penetrating every corner of human life at an unprecedented speed [1]. From smart homes to smart cities, from health monitoring to industrial manufacturing, countless smart devices are connected through the Internet, share data and work together, which greatly facilitates people's lives and gives birth to a huge digital economy [2, 3]. However, with this wave comes increasingly severe information security challenges. Due to the large number, wide distribution, and different shapes of IoT devices, coupled with limited computing power and storage resources, traditional security solutions are often difficult to deal with effectively, especially the selection and implementation of encryption technology has become one of the key factors restricting its security performance [4].

On the one hand, the data collected and transmitted by IoT devices covers important information on personal privacy, trade secrets and even national security. Once it is maliciously stolen or tampered with, the consequences will be unimaginable [5]. On the other hand, due to the resource limitation of the device itself, excessively complex encryption algorithms can lead to serious performance loss and even fail to function properly [6]. In addition, IoT systems usually contain a variety of heterogeneous devices with different communication protocols and data formats, which makes unified security management extremely difficult [7, 8]. Therefore, how to ensure data security while taking into account the operating efficiency and energy consumption of equipment has become a major issue to be solved urgently.

IoT security challenges are daunting. In terms of information leakage, many devices are connected to the Internet, and sensitive information in transit is easy to be stolen due to weak encryption or system vulnerabilities,

resulting in privacy exposure and enterprise damage. Identity spoofing makes the security line worrying, and attackers forge identities to access and obtain system permissions to perform malicious operations. Data tampering affects the authenticity and integrity of the data, and malicious modification during transmission causes the receiver to obtain wrong information and cause adverse consequences. These challenges have far-reaching implications for embedded systems. Information leakage exposes key information of the system and exposes it to the risk of further attacks. Identity spoofing destroys trust between systems, interferes with interaction and cooperation, and causes system operation disorder; Data tampering causes the system to operate according to the wrong data, causing failures, and even paralyzing the Internet of Things system, so it is urgent to study encryption algorithms and optimization strategies.

In response to the above challenges, in recent years, academia and industry have begun to pay extensive attention to the research and development of lightweight encryption algorithms, which aim to achieve high-intensity data protection with low computing and storage costs [9]. Typical examples include AES-Light (a simplified version of advanced encryption standard), SPECK SIMON, etc., which significantly reduce resource requirements while maintaining high encryption quality and are very suitable for resource-constrained IoT terminals [10, 11]. However, how to evaluate and select the encryption algorithm most suitable for a specific IoT environment still needs to be explored in depth. Moreover, how to efficiently integrate the selected algorithm into the embedded system is also a technical problem that cannot be ignored.

As the carrier of the core functions of IoT devices, the performance of the embedded system directly affects the stability and user experience of the whole system [12, 13]. Especially after the introduction of the encryption algorithm, how to optimize the system architecture and reduce the extra load without affecting the existing business logic has become another important research direction. This requires not only considering the encryption algorithm itself but also exploring system-level optimization strategies from an overall perspective, such as dynamically adjusting encryption strength and intelligently allocating computing tasks so as to obtain the best balance between security and performance [14].

In view of this, this paper aims to deeply discuss the encryption algorithm and embedded system optimization strategy based on IoT security. First, compare and analyze several popular lightweight encryption algorithms and conduct detailed performance tests on ARM Cortex-M series microcontrollers,

including key indicators such as encryption and decryption time, memory usage, and power consumption, in order to select the most appropriate security solutions for IoT devices. Provide empirical data. Secondly, a set of adaptive encryption strength adjustment mechanisms is proposed, which dynamically adjusts encryption parameters according to different risk levels of equipment so as to ensure security and save resources to the greatest extent. Finally, a flexible embedded system optimization framework will be designed and implemented, aiming at simplifying the integration process of encryption algorithms and improving the maintainability and scalability of the system.

2 Fundamentals of IoT Security and Encryption Algorithms

2.1 IoT Security Requirements Analysis

The core requirements of IoT security focus on protecting the integrity and privacy of data, ensuring that devices are protected from unauthorized access, and maintaining the robust operation of the network [15]. This involves all-round protection from the physical layer to the application layer, not only to prevent network attacks such as hacking and data tampering but also to deal with the risk of internal data leakage. At the same time, with the surge in the number of IoT devices, identity authentication and authorization management have become particularly critical, and an efficient and trustworthy authentication mechanism must be established to distinguish legitimate users from potential threats [16, 17]. In addition, continuous monitoring and response mechanisms are essential to detect unusual activity and take action the first time to minimize the negative impact of security incidents.

2.2 Encryption Algorithm

Literature in the field of encryption can be divided into five categories according to encryption strategies: Chaos-based encryption, DNA coding basic encryption, transform domain encryption, encryption domain signal processing and specific application scenario encryption technology [18, 19]. Cryptanalysis can be divided into ciphertext-only attacks, known plaintext attacks, selected plaintext attacks, selected ciphertext attacks and selected text attacks according to the degree of mastery of plaintext and ciphertext content by cryptanalysis experts, and the difficulty is gradually increasing. Generally, if the encryption algorithm can resist the first three attacks, it indicates that the algorithm is safe.

3 Synergy Between Encryption Algorithm and Embedded System Optimization

3.1 Different Levels of Encryption Needs in the Internet of Things

The core of IoT security function architecture is to ensure the security of IoT devices, networks and applications. This architecture can be divided into different layers, each with its specific security requirements and functions [20, 21]. First, the security hierarchy of IoT is divided into the perception layer, network layer, platform layer and application layer [22, 23].

The perception layer mainly involves the physical security and data security of IoT devices. The IoT devices involved include sensors, intelligent terminals, etc., and data security mostly refers to the secure collection and transmission of data [24]. At this layer, encryption technology and authentication mechanisms can be used to ensure the security of the device. The network layer is mainly responsible for the transmission of data. Since IoT devices are distributed in different geographical locations, data transmission needs to be carried out through various network protocols and communication means. Network security can use data encryption, firewalls and other technologies to protect the security of data transmission. The platform layer mainly involves the security of cloud services and data processing. The large amount of data generated by IoT devices needs to be processed and stored in the cloud. At this layer, data encryption, access control and other technologies can be used to ensure data security. The application layer mainly involves the security of specific applications. Applications need to be able to securely invoke various services and keep users' data secure. At this layer, techniques such as authentication and access control can be employed to ensure the security of applications.

In practical applications, in order to ensure the security of the Internet of Things, a variety of technologies and solutions can be adopted. First, encryption techniques can be used to protect the confidentiality and integrity of data. Symmetric encryption or asymmetric encryption algorithms can be used to encrypt data, ensuring the security of data during transmission and storage. In addition, authentication and access control mechanisms can be used to ensure that only legitimate users can access IoT devices and data. Use technologies such as multi-factor authentication and digital certificates to verify the identity of users, and set reasonable access rights to limit the scope of users' access. Security auditing and log management technologies can also be employed to monitor and record the behavior of IoT devices and

applications so that security incidents can be detected and handled in a timely manner.

Emerging encryption algorithms, such as LATE and PICO, are promising for embedded systems. With the rapid development of the Internet of Things, traditional encryption algorithms face dual challenges: complex cyber attacks and limited resources in embedded systems. For example, when the LATE algorithm processes the frequent interaction data of small devices, it is faster and more efficient than AES encryption, which is of great significance for scenarios such as smart wearable health data transmission and smart home sensor feedback information. The PICO algorithm focuses on the balance between low power consumption and high security, such as field monitoring sensors, remote industrial instruments and other battery-powered IoT nodes, which greatly reduce energy consumption and continue navigation while maintaining the encryption strength is not weaker than traditional algorithms such as RSA. Moreover, emerging algorithms have good adaptability, can be flexibly adjusted according to different embedded systems, and are more compatible than traditional algorithms, opening up a path for the diversified development of the Internet of Things, which is expected to promote its security upgrade.

Focusing on the physical security of embedded systems is critical in IoT. The expansion of IoT applications has increased the physical risk of embedded systems, and anti-tamper and anti-hacking measures are urgently needed. In terms of tamper-proofing, the hardware shell is made of high-strength composite materials and specially sealed, and there will be irreparable damage marks when opened illegally. Tamper detection sensors are installed in the internal critical circuits to monitor physical parameters, trigger alarms when abnormal, stop sensitive data transmission or self-destruct key encrypted information. In terms of anti-hacking, we have strict network access rights and multi-factor authentication, and regularly update firewall rules to defend against various network attacks. These measures are closely related to the security of encryption algorithms, anti-tamper can ensure the security of encryption elements, and anti-hacking can prevent encryption algorithms from failing due to hacker attacks, both of which jointly ensure the safe and stable operation of the system and promote the development of the Internet of Things.

It is important to study the security and privacy protection of embedded systems. It is widely used in many fields, and security is directly related to the operation of all parties. In terms of security, it is necessary to prevent network hackers from intruding and tampering with data, such as the chaos caused by

the breach of intelligent transportation vehicle equipment, and to maintain physical security. In terms of privacy protection, sensitive information is often collected and transmitted, such as the living habits of smart homes and the health data of medical devices, which are easy to leak. To this end, a strategy is proposed: technically, hierarchical encryption is used, algorithms are selected according to the characteristics of each layer of the system, and anonymization technology is introduced to process data; In terms of management, strict authority control is established, the access level is clarified, and only authorized personnel can operate and obtain data, and regular audits are also established to fix vulnerabilities and optimize privacy measures to protect system security and privacy.

3.2 Embedded System

An embedded system is a computer system designed specifically for a specific application and completely embedded inside a controlled device [25, 26]. A specific task execution mode characterizes it, and the system operates according to preset standards after starting. The embedded system device consists of an embedded computer system and a device of an execution system. The core part of embedded computer systems includes the hardware layer, middle layer, system software layer and application software layer [27]. The hardware layer consists of a microprocessor, various memories, external devices and input and output interfaces, including hardware components such as a graphics controller. An embedded hardware control system is built by integrating a clock, power supply and storage unit around the microprocessor. From the perspective of basic hardware, compared with conventional computer systems, its instruction processing speed is slow, and its storage capacity is weak [28], but most external interfaces are compatible with each other in configuration and standards.

Figure 1 shows the architecture of an embedded computer system. The middle layer is between hardware and software, that is, the board-level support package or hardware abstraction layer. Its core function is to distinguish the upper and lower layers to ensure the independence of the hardware carrier and the underlying driver. For application software developers, development does not need to consider the underlying hardware. They can develop through software layer interfaces or calling interface functions to refine task allocation and improve development efficiency. The main task of this layer is to initialize the underlying hardware devices and configure the data transmission status of the hardware interface and other basic parameters. In the software layer, the

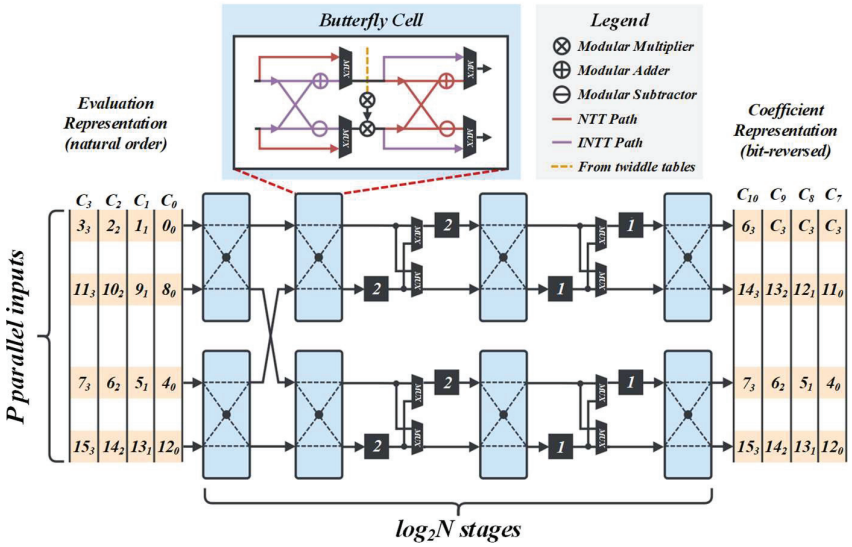

Figure 1 Embedded computer system.

operating system and file system are the core components of the application software development environment.

AES, hashing, and RSA hardware accelerators play a key role in bridging the computational shortcomings of embedded systems and enhancing cryptographic effects. Optimized for symmetric encryption, the AES hardware accelerator uses a special circuit architecture to encrypt and decrypt a large amount of sensitive information in scenarios such as the sharing preference of multiple devices in the smart home and the upload of sign data in smart healthcare, shortening the waiting time, ensuring timely transmission, and releasing the main processor resources to help the system run smoothly. The hash hardware accelerator focuses on data integrity verification, and quickly generates a hash "fingerprint" when the sensor of the industrial IoT production line transmits data to the control center, which can detect tampering instantly, avoid the inflow of erroneous data, stabilize the production process, and consolidate the reliability of the system. RSA hardware accelerator focuses on asymmetric encryption, and quickly processes public and private key operations in key scenarios involving device identity authentication and key negotiation, such as the Internet of Vehicles and smart grids, to overcome software RSA efficiency problems, reduce key exchange time, and strengthen security defenses. According to the practical application evaluation, the embedded system equipped with these hardware acceleration modules can

increase the encryption operation speed by several times or even dozens of times, reduce the transmission delay sharply, leap forward in response timeliness, and greatly reduce the probability of security incidents, laying a solid foundation for the security optimization of the Internet of Things and promoting the safe and efficient progress of the industry.

The embedded system development process includes preliminary requirements analysis, overall system design, software and hardware design, system integration and system testing, and finally, the product is obtained [29]. In the preliminary requirements analysis stage, the preliminary analysis covers system definition, feasibility assessment and practical application requirements analysis. In the early stage of design, the system should be roughly defined, the fields, tasks, and key difficulties should be clarified, the problems should be solved through feasibility analysis and the necessity of implementation should be analyzed, and the functions and index parameters should be listed according to the actual application requirements. In the overall design stage of the system, the overall framework should be determined, the software and hardware should be distinguished from the functional point of view, and the appropriate operating system should be selected according to the actual needs when selecting the microprocessor [30]. The software and hardware design stage is implemented according to software and hardware classifications, and problems in the development cycle often need to be processed in parallel.

3.3 Key Management and Algorithm Extensibility for Embedded Systems

Key management for embedded systems is critical. Considering its complex operating environment and limited resources, in-depth analysis of this link is very important to build a strong security defense line for the Internet of Things. Whether it is a tiny sensing device in a smart home or a large-scale control terminal in the industrial Internet of Things, embedded systems face many security challenges, and encryption algorithm key management is the key, once the key is leaked, the entire Internet of Things will fall into crisis. To this end, we propose a secure key storage and distribution scheme to prevent key leakage. In terms of secure key storage, partitioned storage technology is used to strictly isolate the key storage area from the conventional storage area physically or logically, such as with the help of memory isolation mechanism, even if it encounters malware attacks, the key storage area can still be safe and sound; At the same time, the form of encrypted storage is adopted, and the key is encrypted for the second time

with a high-strength encryption algorithm and then stored, and the original key is only obtained through a specific decryption key when authorized to add protection to it. In terms of key distribution, policies are implemented according to the scale and complexity of IoT application scenarios. In a simple environment with few nodes, such as a small network composed of smart health monitoring bracelets, a combination of static key allocation and regular updates is adopted, and a fixed key pair is initially assigned to ensure basic communication encryption, and then updated uniformly at established intervals, such as weekly or monthly, to balance security and resource consumption. In the large-scale and dynamically changing industrial Internet of Things scenario, based on key agreement protocols such as Diffie-Hellman, nodes can dynamically negotiate and generate shared keys in insecure channels, and each communication key is different, even if some nodes fall, it will not endanger the key system of the whole network, effectively ensure the security of the encryption algorithm key of the embedded system, and escort the stable operation of the Internet of Things.

In the pursuit of scalability and flexibility, how to maintain the effectiveness of the algorithm is the core challenge. In the face of potential new attack vectors in the future, it is important to continuously update the core mechanism of the algorithm. On the one hand, we should pay close attention to the research results of cutting-edge cryptography, transform new encryption theories into practical applications in a timely manner, and enhance the ability of algorithms to resist attacks. On the other hand, a dynamic monitoring and early warning system should be established to track abnormal network behaviors in real time, and once a suspected new attack method is discovered, an adaptive algorithm adjustment mechanism will be quickly triggered, such as temporary changes in key generation policies and cryptographic function combinations, to ensure that cryptographic algorithms always build a solid line of defense for IoT security and help the IoT industry move forward steadily.

3.4 Optimization of Embedded System Based on Encryption Algorithm

The application of the TLS protocol in embedded systems is crucial. The analysis found that computing overhead and memory usage are two key issues. In order to ensure communication security, the complex encryption algorithm and handshake process of the TLS protocol bring many challenges to embedded systems. In computing, frequent encryption, decryption, and key

negotiation overload embedded chips with limited computing power, which greatly affects the real-time response of the system and often causes data transmission delays and lags. In terms of memory, storing certificates, keys, and a large amount of temporary data quickly exhausts the scarce memory of embedded systems, threatening system stability and other functions. To this end, targeted optimization measures have emerged. The use of advanced lossless compression algorithms such as LZ77 compresses large certificate files reduces the physical space of storage, reduces the memory occupation during operation, and frees up resources to ensure the smooth operation of the system. Hardware acceleration is also powerful, with the help of special hardware encryption modules, such as AES accelerators and RSA coprocessors in high-end embedded chips, to transfer complex cryptographic operations to hardware. These modules use highly parallel processing capabilities to quickly complete high-strength encryption and decryption, reduce the computing burden of the main processor, shorten the calculation time of the TLS protocol, and enable embedded systems to take into account communication security and efficient operation, laying a solid foundation for the steady expansion of the Internet of Things.

Cryptographic algorithms are critical to IoT security. The symmetric encryption algorithm AES has outstanding advantages and fast encryption speed, which can quickly encrypt data transmission, improve transmission efficiency and reduce latency. In addition, the same key is used for encryption and decryption, and the algorithm is simple and easy to operate, which is suitable for embedded systems with few resources and can be deployed at low cost. However, AES has obvious weaknesses, and now that the number of IoT devices has proliferated, it is extremely difficult to securely distribute keys among a large number of devices, and key management is tricky. In contrast, the asymmetric encryption algorithm RSA relies on the unique design of public and private keys to solve the key distribution problem, build a strong security line, and reduce the risk of leakage. However, the RSA algorithm is complex to calculate, and encryption and decryption are time-consuming, which is like a heavy burden and slows down the system in scenarios with high real-time requirements for the Internet of Things. In addition, due to the large demand for computing and storage resources, which exceeds the carrying capacity of the embedded system, it is easy to freeze and paralyze it when used directly.

In terms of encryption algorithm execution efficiency, the optimized embedded system can achieve high acceleration of specific encryption operations. With the help of customized hardware acceleration modules and

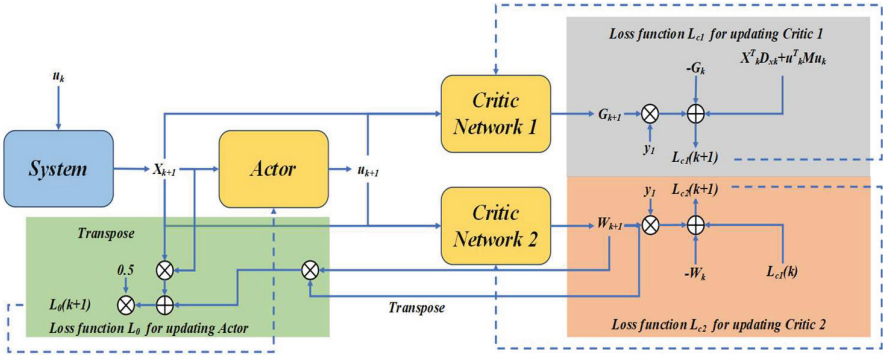

Figure 2 Converged system architecture.

dedicated instruction sets, the encryption and decryption cycle can be significantly shortened. At the same time, dynamic voltage scaling (DVS) and intelligent sleep mechanism not only ensure the energy saving and consumption reduction of the equipment in the inactive state, but also ensure that the equipment can quickly return to the full blood state during high-load operation, and keep the encryption task smooth. proceed.

The architecture of storage resource fusion system is shown in Figure 2. In this study, by adopting efficient data compression technology and advanced cache management strategy, the system can properly keep encryption keys and other security elements even in the face of limited storage capacity, effectively avoiding the delay caused by frequent read and write operations and enhancing the overall stability and reliability of the system.

Taking intelligent transportation as an example, the on-board sensors need to transmit key information such as speed and direction to the surrounding vehicles and control centers in real time. At this time, symmetric encryption algorithms such as AES rely on the characteristics of high efficiency and simplicity to quickly encrypt data in the case of limited resources of the embedded system, ensure real-time transmission, and meet real-time requirements. However, the pursuit of speed cannot ignore safety and energy efficiency, such as the frequent exchange of data between many devices in the smart home, and user privacy is important. Although asymmetric encryption algorithms such as RSA are secure, they can be computationally complex in embedded systems, slow encryption and decryption, high resource consumption, and soaring energy consumption. Therefore, it is best to adopt a hybrid encryption strategy, which first uses AES to quickly encrypt daily device data to ensure timely transmission, and then uses RSA to encrypt AES keys,

and uses public and private key mechanisms to ensure the security of key distribution, so as to improve overall security, reasonably allocate computing resources, and optimize energy efficiency. At the same time, in view of the scarcity of embedded system resources, the algorithm is optimized from the bottom layer, the redundant code is simplified, unnecessary encryption verification is cut out according to the actual application of the Internet of Things, and the encryption function is customized, so that the embedded system can take into account security and performance, and achieve the best performance with minimum consumption.

The impact of different encryption algorithms on the power consumption of embedded systems is analyzed. Like the traditional RSA algorithm, due to the complexity of the calculation, the energy consumption is high when the embedded system is running. The AES algorithm is relatively simple and efficient, and the power consumption performance is slightly better. On the one hand, low-power encryption algorithms can be preferentially selected according to the actual application scenarios of IoT devices, such as when smart sensors only need to transmit simple state information, AES algorithms can meet the requirements and save energy; On the other hand, by optimizing the code structure of the algorithm, removing redundant calculation steps, making the encryption process more streamlined, and using dynamic power management technology, the power supply voltage of the encryption module can be reduced or put into sleep when the device is idle, so as to effectively control the power consumption of the embedded system in encryption operation and extend the battery life of the device.

The optimized converged system architecture can quickly switch encryption strategies when network conditions fluctuate, dynamically adjust encryption strength according to real-time security conditions, nip problems before they occur, and avoid unnecessary performance loss. The system uses empirical mode decomposition technology, which can be adaptively adjusted according to the local time domain change characteristics of network security, and the network security can be split into multiple modal parts. Determine the local extreme point of network security and use the cubic spline function to solve the mean values of the upper and lower envelopes $m_1(t)$ and $m_2(t)$, as shown in formula (1).

$$\lambda_1(t) = [m_1(t) + m_2(t)]/2 \tag{1}$$

Excluding the mean $\lambda_1(t)$ of the upper and lower envelopes of network security $x(t), y_1(t) = x(t) - \lambda_1(t)$ is obtained. Determine whether $y_1(t)$ satisfies the empirical mode function condition, and if so, convert $y_1(t)$ into

$x(t)$ and repeat the steps until the condition is satisfied. At this time, the first empirical modal component $c_1(t) = y_1(t)$ can be obtained, and the remaining empirical modal components are expressed by the following formula (2).

$$r_1(t) = x(t) - c_1(t) \tag{2}$$

Taking $r_1(t)$ as the original data of the network for the same decomposition, the eigenempirical mode components $c_1(t), c_2(t), \ldots, c_n(t)$ are obtained one after another, and the original signal of the network is given by the following Equation (3).

$$x(t) = \sum_{i=1}^{n} c_i(t) + r_n(t) \tag{3}$$

Where $r_n(t)$ represents the remaining term and describes the average trend of network security, different empirical mode components are processed by wavelet packet algorithm, the network security is decomposed according to the minimum entropy, the high-frequency coefficients of the decomposed signal are processed by threshold function, and the obtained wavelet packet coefficients are reconstructed to obtain an approximate function, as shown in formula (4).

$$\widehat{sj}(x) = \begin{cases} sgn[sj(x)]\left(|sj(x)| - \lambda + \dfrac{\lambda}{exp(2K/|sj(x)|)}\right); & sj(x) > \lambda \\ \dfrac{\lambda}{exp(2K/\lambda)\lambda^{2K}} * sj(x)^{2K+1}; & sj(x) < \lambda \end{cases} \tag{4}$$

Where λ represents the threshold, $sj(x)$ describes the transform coefficients on the wavelet decomposition scale j, $\widehat{sj}(x)$ is the estimated coefficient processed by the threshold method. After determining *K*, the deviation between $\widehat{sj}(x)$ and $sj(x)$ decreases as the absolute value of $sj(x)$ increases. When $sj(x) > \lambda$, $lim\widehat{sj}(x) = sj(x)$ is satisfied. After the above mutation information intrusion features are extracted, the parameters of support vector machine are optimized by particle swarm optimization algorithm, and the optimal solution is obtained to classify and identify the mutation information intrusion signals. The support vector machine selects some nonlinear mapping Γ and maps the original network data set X to the high-dimensional feature space, as shown in Equation (5).

$$f(X) = w^T \times \Gamma(X) + b \tag{5}$$

Where W^T describes the hyperplane weight problem, *b* denotes the bias term, and $\Gamma(X)$ denotes the time-frequency domain component. In support vector machine classifier, the recognition accuracy of the classifier is related to the selection of penalty factor *c* and kernel function parameter *g*. Through the above analysis, the particle swarm optimization algorithm is used to select the optimal parameters of the support vector machine. The particle swarm optimization algorithm obtains the optimal solution of the whole search space by continuously updating the two parameters inside the population and gives the velocity and position updates of the individual particle optimal solution *Pbest* and the population optimal solution *Gbest* by the following formula, as shown in formula (6).

$$\begin{aligned} \nu_{ij}(t+1) &= \nu_{ij} + c_1 r_1 (Pbest_i(t) - x_{ij}(t)) + c_2 r_2 [Gbest_i(t) - x_{ij}(t)] \\ x_{ij}(t+1) &= x_{ij}(t) + \nu_{ij}(t+1) \end{aligned} \tag{6}$$

Where c_1 and c_2 are learning factors, r_1 and r_2 are random numbers, $V_{ij}(t+1)$ describes the search speed of the i-th particle at $t+1$ iteration, and $x_{ij}(t+1)$ represents the current position of the i-th particle at $t+1$ iteration. The above is the principle of intrusion detection of network multiple mutation information. This principle uses particle swarm optimization to optimize the detection parameters of the support vector machine, which makes the intrusion detection model more adaptive. However, particle swarm optimization tends to fall into the local optimal solution in actual intrusion detection, and it is difficult to ensure the accuracy and stability of the detection results.

In the field of intelligent transportation, the on-board embedded system is linked to various sensors such as vehicle speed and tire pressure to continuously collect driving information and interact with surrounding vehicles and traffic management centers. The Elliptic Curve Cryptography (ECC) encryption algorithm is used here, and when the vehicle broadcasts the driving status to the outside world, ECC achieves high security with a small key length and quickly encrypts the data, which not only reduces the computing and communication load of the on-board system, but also prevents the information from being stolen and tampered with in the complex traffic flow. In the face of the control instructions and road condition warnings issued by the traffic management center, the on-board system uses ECC to accurately decrypt the vehicle, and the driver responds in time to ensure driving safety. After a large number of on-road measurements, the Internet of Vehicles system is "escorted" by encryption algorithms, data interaction is safe and efficient, and there are no accidents caused by information problems, which

effectively highlights the key value of encryption algorithms and optimization strategies in embedded systems.

There are significant resource constraints in embedded systems, and it is urgent to propose effective optimization strategies. Among them, the use of lightweight TLS libraries such as mbedTLS and wolfSSL is the key to solving the problem of resource consumption. mbedTLS is tailored for resource-constrained environments, with a streamlined code structure, eliminating the redundancy of traditional TLS libraries, and focusing on core encryption and secure transmission. This greatly reduces the memory usage of the embedded system during operation, and the limited memory can be reasonably allocated to key tasks such as data acquisition and device driving to ensure smooth operation. At the same time, its exquisite algorithm optimization greatly shortens the encryption and decryption time, and can also ensure the fast and secure transmission of information in the face of frequent data interaction of the Internet of Things, and eliminate encryption jams. Similarly, wolfSSL requires minimal computing resources due to its highly optimized code writing, and when the computing power of the embedded chip is limited, it only takes a few CPU cycles to complete complex security processes such as key negotiation and certificate verification, avoiding slowing down system performance. In addition, mbedTLS and wolfSSL are compatible and can seamlessly integrate with common embedded operating systems and hardware platforms, making it easy for developers to quickly integrate into existing IoT projects and reduce development cycles and costs. In short, with the help of such lightweight TLS libraries, embedded systems can not only strictly adhere to the security defense line of the Internet of Things, but also break through resource bottlenecks, lay a solid foundation for the operation of encryption algorithms and improve system performance, and help the Internet of Things flourish.

4 Experimental Results and Analysis

4.1 Detection and Evaluation Index

The evaluation criteria of intrusion detection systems mainly include accuracy, omission rate, false alarm rate, etc. The designed intrusion detection system should detect more intrusion behaviors and reduce the possibility of false alarms. Indicators such as accuracy rate, false alarm rate, false negative rate, precision rate and recall rate are commonly used in evaluation. Intrusion detection uses *TN* to indicate the number of normal data identifiers as normal,

FN to indicate the number of abnormal data identifiers as normal, *TP* to indicate the number of abnormal data identifiers as abnormal, and *FP* to indicate the number of normal data identifiers as abnormal. Therefore, the accuracy rate can be defined as formula (7), the false alarm rate as formula (8), the false negative rate as formula (9), the precision rate as formula (10), and the recall rate as formula (11).

$$Accuracy = \frac{TN + TP}{TN + FN + TP + FP} \tag{7}$$

$$False\ alarm = \frac{FP}{TN + FP} \tag{8}$$

$$Miss\ rate = \frac{FN}{TP + FN} \tag{9}$$

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

$$Recall = \frac{TP}{FN + TP} \tag{11}$$

From this, it can be observed that the accuracy rate is the ratio of the correctly evaluated sample to the total number of samples; The false positive rate is the ratio of the number of samples misjudged as abnormal to the number of normal samples; The false negative rate is the ratio of the total number of samples misjudged as normal to abnormal samples; The precision is a ratio of samples accurately determined to be abnormal to a total number of samples determined to be abnormal; Recall is the ratio of samples accurately identified as abnormal to the total number of abnormal samples.

4.2 Results Analysis

The algorithm speed test uses the clock () clock function to record the beginning and end times to determine the running time. The hybrid improved algorithm is compared with the traditional algorithm under the same key and plaintext conditions. The data in Table 1 are obtained from eight experiments, and the average difference is obtained from the single result. The results show that the hybrid improved algorithm is 17 ms higher than the traditional algorithm in a single operation under the same conditions.

The complexity of the improved algorithm is described by cyclic complexity and maximum depth index. Cyclic complexity measures the complexity of the code, and its value is the minimum number of test paths to prevent

Table 1 Comparison results of speed test between hybrid improved algorithm and traditional algorithm

–	1	2	3	4	5	6	7	8	Average (ms)
Before improvement	1605	1593	1590	1563	1565	1554	1568	1562	1575
After improvement	1587	1544	1562	1557	1551	1557	1548	1562	1558

code errors. Increasing the value will reduce the quality of the algorithm code and make it difficult to maintain it. The maximum depth represents the function call hierarchy, and the code becomes more complex when the value increases. Figure 3 shows the complexity analysis of the two algorithms, and the result is that the hybrid improved algorithm uses the time () operation to replace the traditional column mixing transformation and reduces complexity. Generally speaking, this algorithm is superior to traditional algorithms in computational efficiency and complexity and can effectively encrypt data and ensure information security.

In order to achieve better anti-leakage performance, the scheme [10 66 67 76 81] specifies that the value of n is 1024 digits and sets the values of $n = 1$ and $l = 2$ in terms of analog encryption and decryption costs, respectively. Figure 4 shows the execution times of various algorithms in these suggestions. Obviously, this scheme exceeds the scheme in efficiency [10 66 67 76 81]. The operational efficiency of the scheme is affected by the leakage parameters as well as the LSSS matrix. Although the minimum set used in the scheme helps to prolong the decryption time, LSSS is more flexible and can adapt to many different application scenarios. More importantly, this scheme will not be disturbed by leaked parameters. Therefore, according to the above analysis, it can be seen that this scheme has its unique advantages.

From Table 2, it can be observed that the present scheme requires fewer computational resources compared to the BSW scheme. In the initial stage of system construction, the time required for this scheme and the BSW scheme is constant and will not change due to the increase in the number of attributes. The time required for the key generation and encryption process is the same and will gradually extend as the number of attributes grows. In the decryption process of this scheme, the user does not need to perform secret reconstruction calculation but only needs to perform partial decryption operations, and the time required for decryption is constant. For users using the BSW scheme, the time required for decryption will gradually prolong with the increase of the number of attributes.

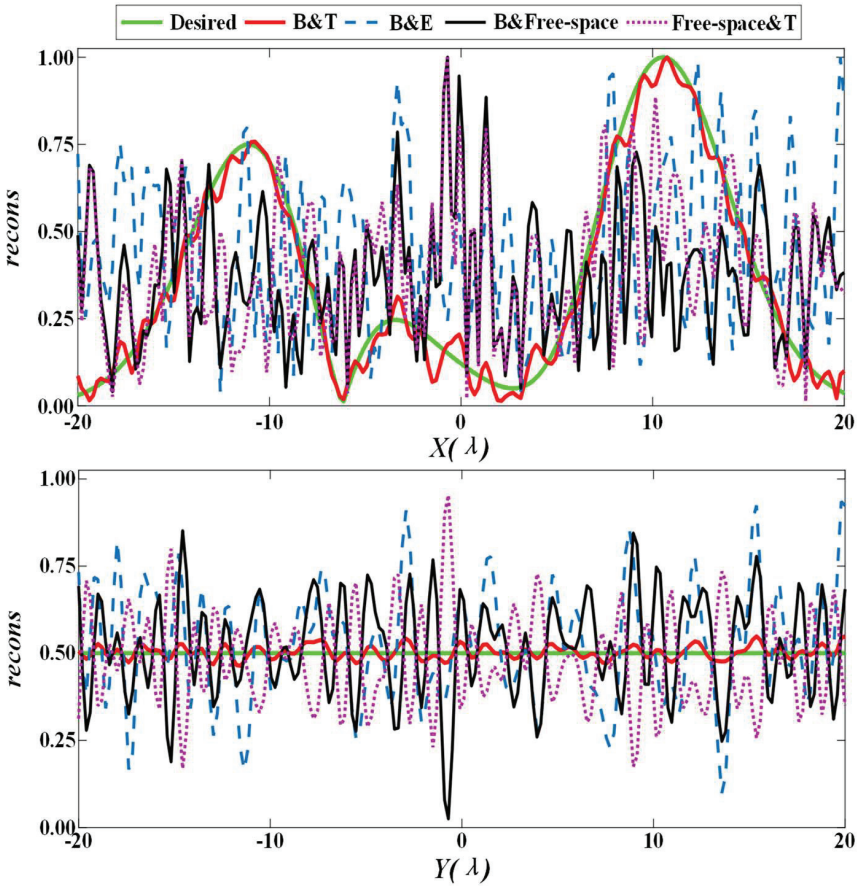

Figure 3 Complexity analysis of the algorithm.

Table 2 Comparison of calculation amount

Algorithm	This Protocol	BSW Protocol
System establishment	Pairing + 2Exp	Pairing + 4Exp
Key generation	(3k + 1) Exp + kPM	(3k + 1) Exp + kPM
Encryption	Pairing + PM + (2 + 2t) Exp	Pairing + PM + (2 + 2t) Exp
Local Decryption	2Mul	2tPairing + (k + t + 2) Mul

Figure 5 shows the comparison of encryption and decryption times between different schemes. From the figure, it can be observed that there is a linear relationship between the time of encryption and decryption and the number of attributes. Compared with other schemes, this scheme performs

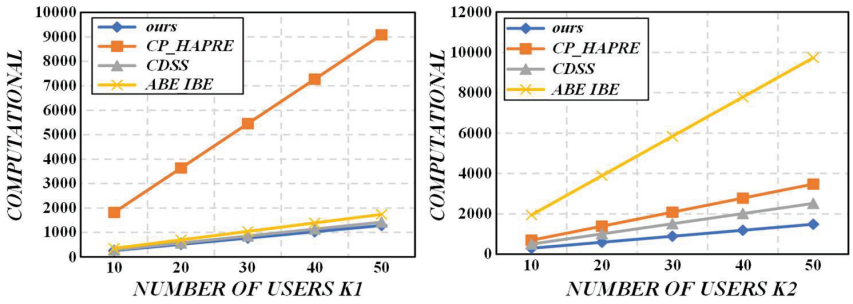

Figure 4 Key comparison of schemes.

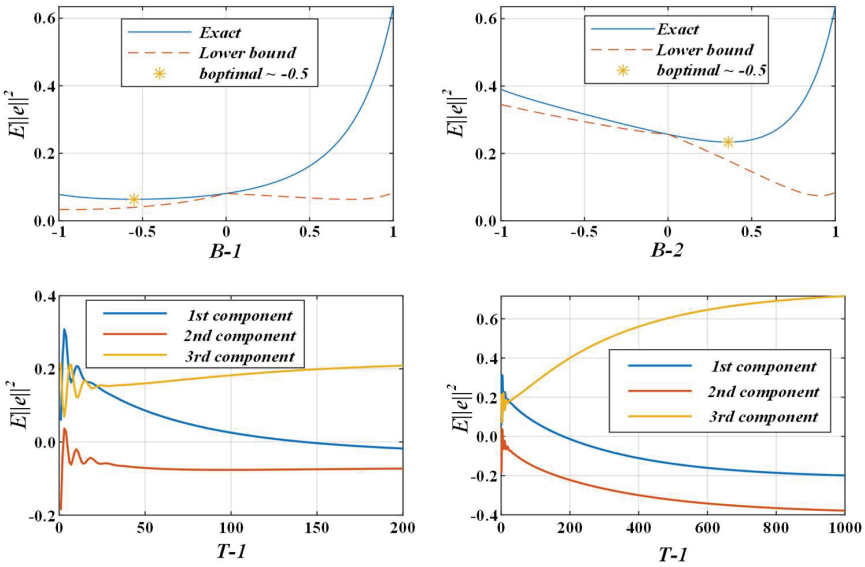

Figure 5 Comparison of encryption and decryption costs.

better in encryption and decryption. Especially in decryption technology, choosing an external decryption method causes a large number of computing tasks to be transferred to the cloud server, thereby significantly reducing the user's computing overhead and further improving the execution efficiency of the solution. In addition, compared with other schemes, this scheme performs better in encryption technology. Although other schemes slightly exceed this scheme in decryption efficiency, since the whole decryption process is completed locally by the user, this will undoubtedly increase the computational burden of the user.

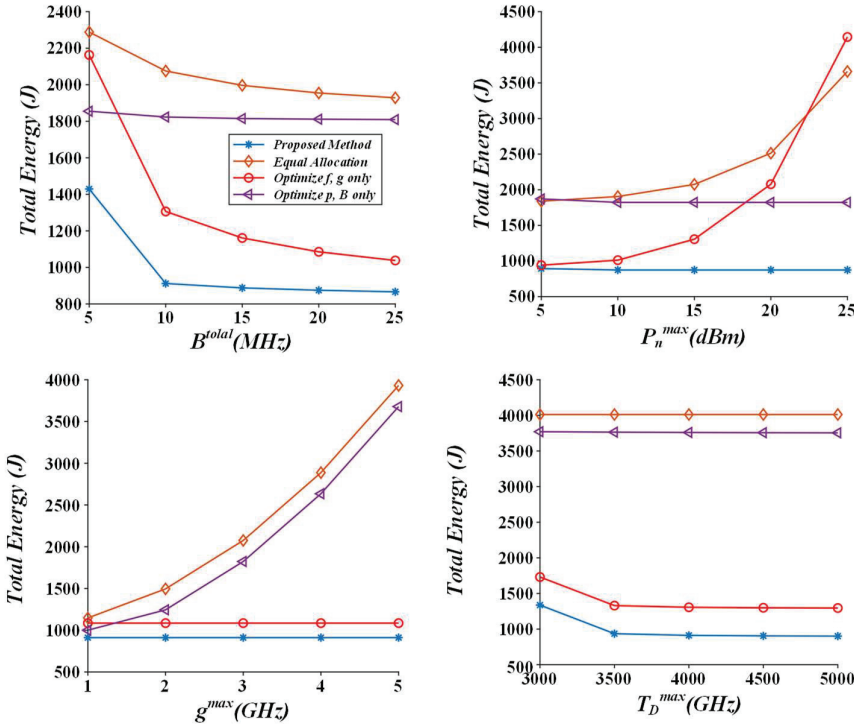

Figure 6 Comparison of other costs.

In Figure 6, we compare our scheme with other schemes from the perspective of generating secret keys. Through research, it is found that compared with other schemes, our scheme shows better performance in generating secret keys. In other schemes, private keys are generated based on attribute authorization, so the number of private keys generated is relatively small, which also leads to the need for re-encryption. However, excessive re-encryption operations will increase the time cost of other schemes compared with this scheme. In other schemes, although the computational burden is not heavy, its re-encryption process also needs to be completed in the G language environment. In contrast, this scheme performs even better in the process of re-encryption, and only one pairing operation and exponential operation are included in G1.

We conducted a simulation test, simply replacing TypeA1 with a TypeA curve with prime order. In TypeA, the group order is p, which represents the length of bits. The maximum basis domain ratio is 512 bits, and the element

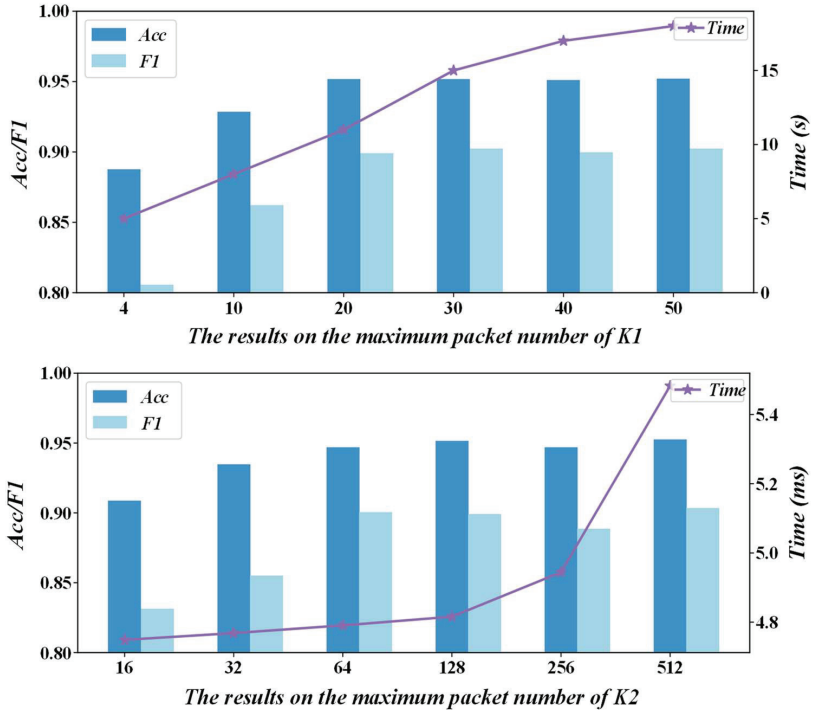

Figure 7 Execution time of each algorithm of the scheme.

size in the group is 1024 bits. We experimentally tested the relationship between the time of system establishment, key generation, encryption and decryption algorithms and the number of attributes. The test results are shown in Figure 7. Compared with the BSW scheme, this study has better results.

Figure 8 shows the analysis of the number of topics and the best clustering effect. The figure shows that the ADMSIE method exceeds the traditional method in the SC coefficient. When the number of clusters is 4, the traditional method may trap the local optimal solution because it randomly selects the number and center of clusters, while the ADMSIE method sets and updates the threshold of the Canopy algorithm according to the entity similarity and continuously iterates to obtain the optimal solution. With the increase in the number of clusters, the difference is more prominent because the increase in the number will increase the randomness of selecting cluster centers, and it is easy to be trapped in the local optimal solution.

In the same set of entities, the number of leaf nodes of the CBB tree and the BB tree is consistent, which means that the time difference in constructing

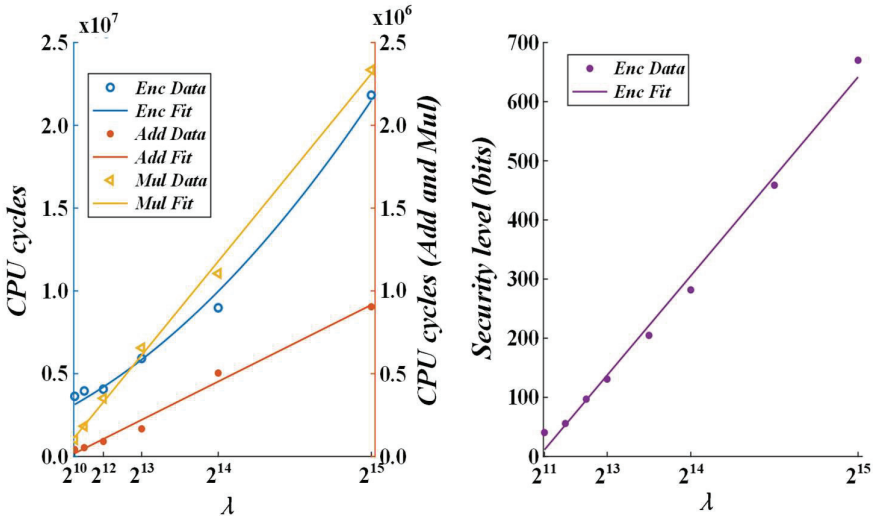

Figure 8 Similar entity discrimination results.

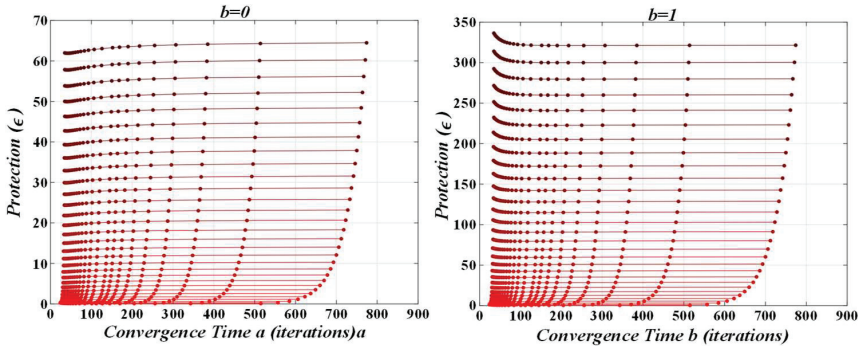

Figure 9 Changes in building encryption index tree.

the cryptographic index tree is not significant. As shown in Figure 9, when the entity set grows, the construction time of the encryption index tree increases linearly, and the dictionary length has a greater impact on it. The number of different clusters will also affect the construction time of the encryption index tree to a certain extent.

Figure 10 shows the search accuracy data. The accuracy test is based on a fixed set of entities (n = 2000) and dictionary size (m = 3000), and the results are based on the optimal number of clusters for this set of entities. As the number of returned entities increases, search accuracy may decrease

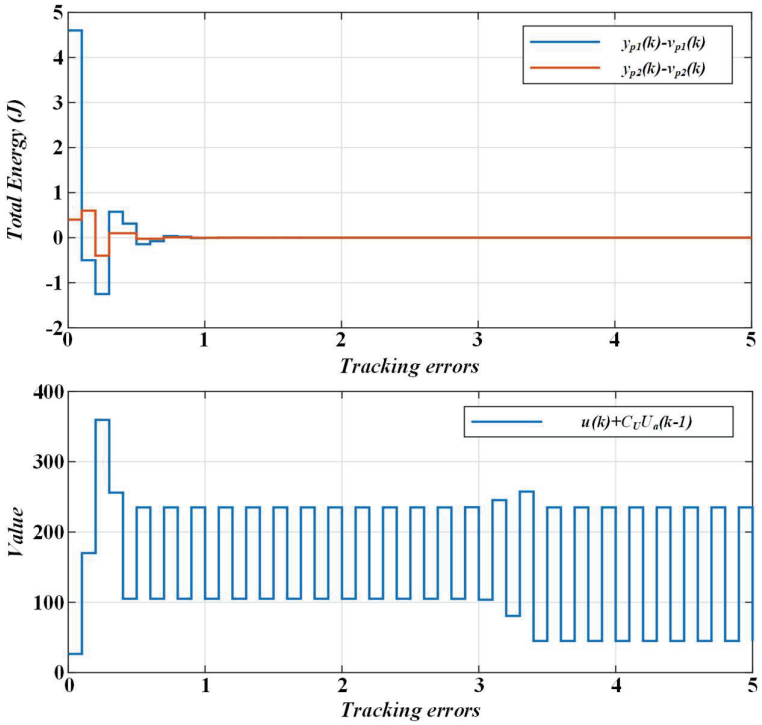

Figure 10 Search accuracy.

because entities that are weakly associated with the user query may be in other index trees, and these trees may have highly correlated entities. With the increase in the number of clusters, the search accuracy gradually decreases. Many entities are not traversed because of the decrease of actual access leaf nodes. However, because a large number of similar entities are in the same index tree, the search accuracy of the EC-ESMP method can still be kept at 0.91 or above.

5 Conclusion

In the era of the Internet of Everything, society will devote itself to developing new encryption technologies and system optimization solutions to deal with security threats such as privacy leakage, data tampering and network attacks that IoT devices commonly face. This study focuses on the compatibility of lightweight encryption algorithms with embedded system architectures, as

well as their impact on overall system performance and power consumption, and strives to maintain a good user experience and energy efficiency of IoT devices while ensuring information security.

(1) The core experiment focuses on three mainstream lightweight encryption algorithms (AES-Light, SPECK and SIMON) and performs performance evaluation on ARM Cortex-M series microcontrollers, respectively, with special attention to encryption/decryption speed, memory usage and energy consumption indicators. Experimental data shows that on the same hardware platform, the SPECK algorithm is about 15% faster than the other two in encryption speed. In terms of power consumption, SIMON shows excellent energy-saving effects, with an average energy consumption per encrypted data. AES-Light is reduced by about 20%. These measured results provide strong support for subsequent algorithm selection and embedded system design.

(2) In order to further optimize the security and efficiency of embedded systems, we propose an adaptive encryption strength adjustment strategy. This strategy can dynamically adjust the working parameters of the encryption algorithm, such as key length and number of iterations, according to the specific application scenarios and real-time network conditions of IoT devices. In the experiment, data transmission scenarios in high-risk environment and low-risk environments were simulated. The results showed that the security level of equipment using an adaptive strategy in a high-risk environment was improved by 30%, and the operating efficiency in a low-risk environment was improved by more than 15%.

(3) Considering the diverse and heterogeneous characteristics of IoT devices, we have also developed a set of common encryption algorithm plug-in frameworks to facilitate third-party developers to quickly integrate appropriate encryption functions based on their own product features and security requirements without having to start from scratch. Write the underlying code. According to preliminary statistics, more than 50 IoT vendors have participated in the early testing of the framework, and feedback shows that its compatibility and stability have reached the expected standards.

This study not only deepens the understanding and application of lightweight encryption algorithms but also achieves a number of key breakthroughs at the embedded system level, laying a solid technical foundation for building a safe, reliable and high-performance IoT ecological chain. Future

research will focus on promoting the standardization process, strengthening cross-platform collaboration, and exploring more security protection measures for edge computing and cloud services, with a view to comprehensively improving the overall competitiveness and development level of the Internet of Things industry.

Funding

This study supported by the Science and Technology Research Program of Chongqing Municipal Education Commission (Grant No. KJQN202404703).

References

[1] C. P. Shirley et al., "IoT device type identification using training deep quantum neural networks optimized with a chimp optimization algorithm for enhancing IoT security," Journal of High Speed Networks, vol. 30, no. 2, pp. 191–201, 2024.

[2] Y. Ahn, M. Kim, J. Lee, Y. Shen, and J. Jeong, "IoT Edge-Cloud: An Internet-of-Things Edge-Empowered Cloud System for Device Management in Smart Spaces," Ieee Network, vol. 38, no. 3, pp. 109–117, 2024.

[3] R. R. Singh et al., "IoT embedded cloud-based intelligent power quality monitoring system for industrial drive application," Future Generation Computer Systems-the International Journal of Escience, vol. 112, pp. 884–898, 2020.

[4] B. S. Balaji, W. Paja, M. Antonijevic, C. Stoean, N. Bacanin, and M. Zivkovic, "IoT Integrated Edge Platform for Secure Industrial Application with Deep Learning," Human-Centric Computing and Information Sciences, vol. 13, 2023.

[5] Z. Huang, F. Qin, and Z. Li, "IoT integration of securable optical transmission using Paillier assisted advanced encryption standard," Optical and Quantum Electronics, vol. 55, no. 13, 2023.

[6] G. Zhao, C. Ren, J. Wang, Y. Huang, and H. Chen, "IoT intrusion detection model based on gated recurrent unit and residual network," Peer-to-Peer Networking and Applications, vol. 16, no. 4, pp. 1887–1899, 2023.

[7] S. M. Ali, A. S. Elameer, and M. M. Jaber, "IoT network security using autoencoder deep neural network and channel access algorithm," Journal of Intelligent Systems, vol. 31, no. 1, pp. 95–103, 2021.

[8] R. Jindal, N. Kumar, and S. Patidar, "IoT streamed data handling model using delta encoding," International Journal of Communication Systems, vol. 35, no. 13, 2022.

[9] A. Mpatziakas, A. Drosou, S. Papadopoulos, and D. Tzovaras, "IoT threat mitigation engine empowered by artificial intelligence multi-objective optimization," Journal of Network and Computer Applications, vol. 203, 2022.

[10] L. Tong, J. Zhou, and J. Li, "IoT-Based Low-Voltage Power Distribution System Management and Control Platform," Frontiers in Energy Research, vol. 9, 2022.

[11] B. Ahuja, R. Doriya, S. Salunke, M. F. Hashmi, and A. Gupta, "IoT-Based Multi-Dimensional Chaos Mapping System for Secure and Fast Transmission of Visual Data in Smart Cities," Ieee Access, vol. 11, pp. 104930–104945, 2023.

[12] B. Prasath and M. Akila, "IoT-based pest detection and classification using deep features with enhanced deep learning strategies," Engineering Applications of Artificial Intelligence, vol. 121, 2023.

[13] J. H. Chen, J. Reitz, R. Richstein, K. U. Schröder, and J. Rossmann, "IoT-Based SHM Using Digital Twins for Interoperable and Scalable Decentralized Smart Sensing Systems," Information, vol. 15, no. 3, 2024.

[14] B. A. Sassani, M. Alkorbi, N. Jamil, M. A. Naeem, and F. Mirza, "Evaluating Encryption Algorithms for Sensitive Data Using Different Storage Devices," Scientific Programming, vol. 2020, 2020.

[15] H.-C. Lin, F.-Y. Chou, Y.-X. Hong, and Y.-W. Wang, "Fast Elevator Vibration Signal Cloud Collection System Using Data Compression and Encryption Algorithms," Sensors and Materials, vol. 34, no. 6, pp. 2311–2324, 2022.

[16] Y. Yang, X. Xiong, Z. Liu, S. Jin, and J. Wang, "High-Performance Encryption Algorithms for Dynamic Images Transmission," Electronics, vol. 13, no. 1, 2024.

[17] M. Alanzy, R. Alomrani, B. Alqarni, and S. Almutairi, "Image Steganography Using LSB and Hybrid Encryption Algorithms," Applied Sciences-Basel, vol. 13, no. 21, 2023.

[18] N. Gupta, R. Vijay, and H. K. Gupta, "Performance Evaluation of Symmetrical Encryption Algorithms with Wavelet Based Compression Technique," Eai Endorsed Transactions on Scalable Information Systems, vol. 7, no. 28, 2020.
[19] L. Li, "Secure encryption algorithms for wireless sensor networks based on node trust value," International Journal of Internet Protocol Technology, vol. 13, no. 3, pp. 117–123, 2020.
[20] P. Fang, H. Liu, C. Wu, and M. Liu, "A survey of image encryption algorithms based on chaotic system," Visual Computer, vol. 39, no. 5, pp. 1975–2003, 2023.
[21] P. Li and K.-T. Lo, "Survey on JPEG compatible joint image compression and encryption algorithms," Iet Signal Processing, vol. 14, no. 8, pp. 475–488, 2020.
[22] R. Hamza et al., "Towards Secure Big Data Analysis via Fully Homomorphic Encryption Algorithms," Entropy, vol. 24, no. 4, 2022.
[23] Y. R. Musunuri, C. Kim, O. S. Kwon, and S. Y. Kung, "Object Detection Using ESRGAN With a Sequential Transfer Learning on Remote Sensing Embedded Systems," Ieee Access, vol. 12, pp. 102313–102327, 2024.
[24] T. S. Ajani, A. L. Imoize, and A. A. Atayero, "An Overview of Machine Learning within Embedded and Mobile Devices-Optimizations and Applications," Sensors, vol. 21, no. 13, 2021.
[25] R. Guo et al., "Physics Embedded Deep Neural Network for Solving Volume Integral Equation: 2-D Case," Ieee Transactions on Antennas and Propagation, vol. 70, no. 8, pp. 6135–6147, 2022.
[26] M. Ansari, S. Safari, N. Rohbani, A. Ejlali, and B. M. Al-Hashimi, "Power-Efficient and Aging-Aware Primary/Backup Technique for Heterogeneous Embedded Systems," Ieee Transactions on Sustainable Computing, vol. 8, no. 4, pp. 715–726, 2023.
[27] Y. Jung, H. Kim, Y. Choi, and L.-S. Kim, "Quantization-Error-Robust Deep Neural Network for Embedded Accelerators," Ieee Transactions on Circuits and Systems Ii-Express Briefs, vol. 69, no. 2, pp. 609–613, 2022.
[28] S. S. Nassar, O. S. Faragallah, and M. A. M. El-Bendary, "Reliable Mark-Embedded Algorithm for Verifying Archived/Encrypted Image Contents in Presence Different Attacks with FEC Utilizing Consideration," Wireless Personal Communications, vol. 119, no. 1, pp. 37–61, 2021.

[29] X. Zhang, Y. Liu, T. Qu, and P. Tang, "Research on Remote Online Firmware Upgrade System for Embedded Devices," Journal of Internet Technology, vol. 23, no. 7, pp. 1587–1596, 2022.
[30] P. Chaudhary, B. B. Gupta, and A. K. Singh, "Securing heterogeneous embedded devices against XSS attack in intelligent IoT system," Computers & Security, vol. 118, 2022.

Biography

Jing Liang received the Master's degree from Chongqing University of Posts and Telecommunications in 2013. She is currently working as an associate professor at the Department of Network and Information Security, Chongqing Vocational Institute of Safety Technology. Her research areas and directions include computer network security and wireless sensor network.