Research on a Lightweight SM4 Algorithm-Driven Secure Communication Mechanism for Communication Links

Xiaoli Tang

School of Electronic Engineering and Optoelectronic Technology, Nanjing University of Science and Technology ZiJin College, Nanjing, 210023, China
E-mail: Tangxl17@outlook.com

Received 07 January 2026; Accepted 23 March 2026

Abstract

Secure communication for resource-constrained embedded devices must simultaneously satisfy cryptographic compliance, low-latency transmission, and robustness against link disturbances, which makes lightweight deployment of national cryptographic algorithms a critical issue in edge and industrial communication systems. To address this problem, a lightweight SM4-based communication security mechanism is developed for heterogeneous embedded links. The proposed method integrates S-box structure compression, round-function optimization, key-scheduling reconstruction, module trimming, and interface adaptation to construct an efficient encryption/decryption framework that supports UART, CAN, and BLE communication environments. The significance of this study lies not only in reducing the computational and storage burden of SM4, but also in enabling practical deployment of a national-standard encryption mechanism in low-power, real-time, multi-interface embedded platforms. Experimental results on the STM32L432KC platform show that, under the BLE link, the proposed scheme achieves an average encryption latency of 165.8 μs and a throughput

Journal of Cyber Security and Mobility, Vol. 15_2, 415–442.
doi: 10.13052/jcsm2245-1439.1526

of 8.42 KB/s. Compared with existing SM4 and AES implementations, the method provides better delay-throughput performance while maintaining strong anti-interference capability and state-recovery resilience under multiple attack and error-injection scenarios. These results indicate that the proposed mechanism is suitable for secure communication tasks requiring both lightweight implementation and link-level adaptability in embedded systems.

Keywords: Lightweight encryption, communication security mechanism, SM4 optimization, embedded system, link adaptation.

1 Introduction

Recent studies have shown that secure communication for edge devices, IoT nodes, and embedded industrial terminals must simultaneously satisfy low latency, low power consumption, and resistance to multiple attack surfaces. Chen, Li, and Zhou (2022) emphasized that lightweight and high-performance data protection is essential for edge network security, especially when data confidentiality and transmission efficiency must be balanced under constrained resources [1]. Goulart, Chennamaneni, Torre, et al. (2022) further reviewed wide-area IoT networks and pointed out that lightweight security mechanisms are no longer optional add-ons, but a necessary foundation for scalable and reliable distributed communication [2]. In application-oriented scenarios, Justindhas and Jeyanthi (2023) demonstrated that lightweight cryptography can effectively support real-time monitoring systems, yet their model mainly focused on cloud-connected sensing environments rather than heterogeneous embedded links [3]. Wang (2024) and Zhang, Meng, and Wang (2025) also confirmed that current network security research increasingly values adaptive optimization, intelligent protection, and implementation efficiency, but these studies are more oriented toward vulnerability detection or security-system optimization, rather than lightweight national-cryptography deployment on resource-constrained platforms [4].

Against this background, the significance of this study lies in bridging the gap between cryptographic compliance, embedded deployability, and heterogeneous communication adaptability. Different from existing models, the proposed method is not limited to a generic lightweight encryption design. It integrates lightweight SM4 optimization with communication-link security mechanisms, and supports UART, CAN, and BLE interfaces through modular

adaptation. This design is particularly relevant to readers interested in embedded security, industrial communication, and lightweight protection for domestic cryptographic systems. Compared with the models reported in the literature, the proposed scheme provides stronger compatibility with national cryptographic standards, lower deployment overhead, better link-level adaptability, and more stable encryption/decryption performance under practical embedded constraints. The key novelty of this study lies in the coordinated design of a lightweight SM4 algorithm and a link-oriented secure communication mechanism for heterogeneous embedded environments. Unlike conventional studies that only optimize cipher execution or only add link-layer protection, the proposed method integrates three aspects into a unified framework: compact SM4 reconstruction through S-box compression, round-function simplification, and key-scheduling restructuring; communication-security enhancement through integrity verification, replay protection, and link-state-aware adaptation; and engineering-oriented deployment validation across UART, CAN, and BLE interfaces on the STM32L432KC platform. Therefore, the contribution of this work is not a single algorithmic refinement, but a complete embedded security model that jointly addresses cryptographic compliance, real-time execution, resource limitation, and multi-link adaptability.

2 Requirements Analysis for Communication Link Security Mechanisms

In modern heterogeneous communication links with multiple nodes and high concurrency, the security mechanism must satisfy several essential requirements, including confidentiality, message integrity, replay resistance, and low-latency execution under strict resource constraints. Recent studies have shown that lightweight security mechanisms must consider not only cryptographic strength but also deployment feasibility on embedded platforms with limited storage and computing capability. Kumar et al. (2024) pointed out that lightweight security design for resource-constrained IoT devices must carefully balance security protection with memory footprint and processing overhead [6]. El Hanine et al. (2025) further demonstrated that when lightweight cryptographic algorithms are deployed on embedded platforms such as ESP32 and STM32, both timing efficiency and hardware adaptability become critical evaluation indicators [7].

For communication-link-oriented systems, security requirements are not limited to encryption alone. Le et al. (2025) integrated confidentiality,

integrity verification, and replay protection into the Modbus TCP communication environment, indicating that application-layer security mechanisms are necessary for legacy communication links [8]. Wang et al. (2025) also showed that in in-vehicle CAN networks, dynamic key management, authentication freshness, and real-time communication performance must be jointly considered to avoid increased bus load and potential security vulnerabilities [9]. These studies indicate that an effective communication link security mechanism should integrate encryption, authentication, integrity verification, and replay detection. Therefore, the lightweight SM4-based communication link protection scheme proposed in this work aims to achieve secure, low-latency, and resource-efficient communication in heterogeneous embedded environments.

3 Lightweight Improvement Design of the SM4 Algorithm

3.1 Fundamental Principles of the SM4 Algorithm

SM4 is a kind of symmetric block cipher used in Chinese commercial cryptographic standards. It has a 128-bit block and key length. The core of the algorithm consists of 32 rounds of non-linear transformation, including circular key addition, S-box replacement, and linear operation. However, the S-box lookup, 32-round depth, and the complexity of key expansion in an embedded or resource constrained system put a heavy burden on storage and computing resources. This provides a structure entry point for future lightweight designs that include lookup compression, parallel simplification, and circular function reconstruction.

3.2 Computational Optimization Strategies and Complexity Reduction

To decrease the complexity of SM4 in embedded environment, this paper presents an approach to simplify the structure of S-box, linear transform and key extension. For S-box operations, a compact search compression policy was used, which mapped the original 8×8 bit substitution logic to 4×4 high/low bit separation[6]. This, in combination with a parallel lookup path, effectively reduces memory access latency. For the linear transformation stage, the original SM4 linear transform is expressed as Equation (1):

$$L(B) = B \oplus (B <<< 2) \oplus (B <<< 10) \oplus (B <<< 18) \oplus (B <<< 24) \tag{1}$$

On this basis, the lightweight sparse reconstruction adopted in this study is further written as Equation (2):

$$L'(B) = B \cdot P \bmod 2^{32} \tag{2}$$

where B is a 32-bit intermediate variable, $<<<\ 10$ denotes a cyclic left shift operation, and P represents the sparse matrix structure of the linear transformation weights. This sparse design significantly reduces the number of logic gates. $\bmod 2^{32}$ indicates a 32-bit truncation operation, ensuring data width alignment with embedded systems. Additionally, round key expansion employs a hard-coded strategy pre-configuring 16 rounds of key scheduling tables, effectively avoiding runtime dynamic key computation. This provides a framework for the subsequent lightweight module level implementation.

3.3 Code-Level Lightweight Implementation and Module Reconstruction

Based on structure simplification and circular function refactoring, SM4 is implemented with modular and bitwise operation fusion strategy, which is suitable for resource constrained embedded environment. In order to reduce memory access rate, S-table lookups are designed as dual-channel 4×4 bit register mapping, which avoids the overhead of full table storage. Algorithm 1 presents the optimized round function pseudocode:

Algorithm 1 Lightweight SM4 round function pseudocode

```
// SM4 round function (lightweight version)
uint32_t L_transform(uint32_t B) {
    return B ^ ROTL(B, 2) ^ ROTL(B, 10) ^ ROTL(B, 18) ^ ROTL(B, 24);
}
uint32_t T(uint32_t input) {
    uint8_t b0 = SBOX[input >> 24];
    uint8_t b1 = SBOX[(input >> 16) & 0xFF];
    uint8_t b2 = SBOX[(input >> 8) & 0xFF];
    uint8_t b3 = SBOX[input & 0xFF];
    uint32_t B = (b0 << 24) | (b1 << 16) | (b2 << 8) | b3;
    return L_transform(B);
}
```

Where ROTL (x, n) is a cyclic left-shift of x by n bits; SBOX is a 4×4 search table constant, T () is a master wheel function that combines

non-linear and linear transformations, and the overall configuration allows RAM utilization to be minimized by register-level access while maintaining balanced instruction execution and encryption/decryption rates on low frequency MCUs. This module provides a portable, trimmable functional level implementation for later embedded deployment targeting link security mechanisms.

3.4 Energy Consumption and Storage Space Optimization Design

In the light weight SM4 implementation, the structure level optimization of power consumption and storage is necessary to ensure the long-term on-line operation of the equipment in the communication link and the efficient use of the storage resources [10]. To quantitatively describe the joint influence of power consumption, register occupation, ROM table size, and I/O access frequency, the unified resource-cost function established in this study is given in Equation (3):

$$C_{\text{total}} = \alpha \cdot P_{\text{op}} + \beta \cdot \mathrm{M}_{\text{reg}} + \gamma \cdot S_{\text{rom}} + \delta \cdot L_{\text{io}} \tag{3}$$

Where C_{total} represents the total resource cost, P_{op} denotes the average power consumption per round function operation, M_{reg} indicates the number of bytes used in registers, S_{rom} reflects the space occupied by S-boxes and key tables in ROM, L_{io} measures the number of I/O accesses, and $\alpha, \beta, \gamma, \delta$ is the platform-specific weighting coefficient. The model parameters are shown in Table 1, in order to accommodate different micro-controllers. During the power consumption assessment, the platform reference curves are called as shown in Figure 1, which show the trend of total energy consumption for each module level pruning scheme.

Table 1 presents the resource configuration of the core modules in the lightweight SM4 implementation. The listed occupied size indicates the memory or operation-level cost of each module after optimization. The S-box lookup unit is compressed from 256 to 64 bytes through high/low-bit splitting and shared lookup, which directly reduces the storage burden of nonlinear substitution. The register allocation and I/O intermediate buffer entries reflect the effort to shorten instruction paths and reduce memory access frequency during round execution. In addition, the power-consumption model coefficients are not fixed constants, but platform-related weighted parameters, which means that the lightweight strategy is designed to remain adaptable under different embedded hardware conditions.

Table 1 Module resource parameter configuration in lightweight SM4 implementation

Module Name	Parameter Symbol	Occupied Size (Bytes/Operation)	Optimization Strategy
S-box Lookup Unit	S_{rom}	256→64	High/Low Bit Splitting + Shared Compressed Lookup
Register Allocation	M_{reg}	128	Simplified Function Mapping by Round
I/O intermediate buffer	L_{io}	56	Reduced Memory Access Counts + Short Instruction Path Mapping
Power consumption model coefficients	α, β	Refer to experimental platform parameters	Dynamically weighted based on hardware platform

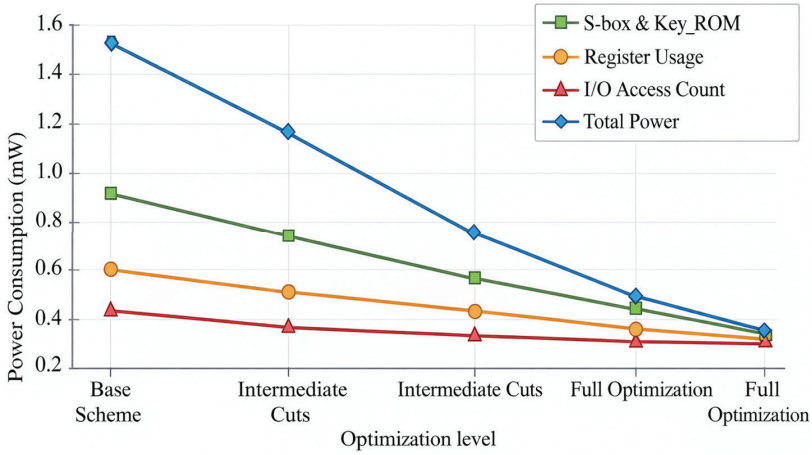

Figure 1 Power consumption contribution curves of different modules before and after lightweight optimization.

Structurally, the algorithm is designed to reduce the number of memory access via module pruning and lookup table diversion, at the same time to optimize the size of the code resident block. This achieves progressive convergence in power consumption and compressed storage deployment, providing efficient algorithm support for embedded link encryption modules [11].

3.5 Security vs. Lightweight Balance Assessment

During the lightweight design of the SM4 algorithm, establishing a clear evaluation balance mechanism between security strength and computational resource consumption is essential to ensure its deployability on resource-constrained platforms [16]. To measure the trade-off between security retention and resource consumption under lightweight design, the composite evaluation function is defined as Equation (4):

$$F_{\text{balance}} = \lambda_1 \cdot R_{\text{diff}} + \lambda_2 \cdot S_{\text{conf}} - \lambda_3 \cdot C_{\text{comp}} - \lambda_4 \cdot M_{\text{mem}} \tag{4}$$

Where R_{diff} denotes the differential diffusion strength of the rounding function, S_{conf} represents the confusion entropy generated after S-box substitution, C_{comp} indicates computational complexity under the compressed structure, M_{mem} signifies total storage overhead, and $\lambda_1 \sim \lambda_4$ serves as a platform-related adjustment coefficient for dynamically balancing performance objectives. Furthermore, to characterize side-channel tolerance and equivalence degradation after structural simplification, the discrimination function is introduced as Equation (5):

$$\Delta_{\text{sec}} = |H(S_{\text{std}}) - H(S_{\text{lite}})| + \in \cdot \frac{T_{\text{std}} - T_{\text{lite}}}{T_{\text{std}}} \tag{5}$$

where $H(S)$ denotes the information entropy function of the S-box, $\in$ represents the platform-preset error tolerance coefficient, and T_{std} and T_{lite} denote the single-round processing delays for standard and lightweight architectures, respectively. This measure measures how the simplified structure affects the overall encryption strength and performance [13, 14]. In order to support an engineering configuration for a lightweight strategy, Table 2 lists the resource constraints for different communication links. These parameters are used to guide the lightweight design of the encryption module.

Table 2 summarizes the resource constraints associated with different communication links, including flash availability, RAM allocation, maximum latency tolerance, and communication bandwidth. These parameters represent the practical hardware limitations that must be considered when deploying cryptographic modules on embedded devices. For example, the flash and RAM values determine the maximum size of the encryption module and the buffering strategy used during data transmission. The latency constraint reflects the real-time requirement of each link type, which directly influences the scheduling of encryption and decryption operations. In addition, the bandwidth parameter indicates the achievable data throughput under

Table 2 Mapping of lightweight SM4 design parameters to security dimensions

Module Parameter	Control Symbol	Corresponding Security Dimension	Optimization Strategy Description
Number of Iterations for Round Function	ρ	Differential Diffusion Intensity	Limited to $\geq$16 iterations to maintain nonlinear coverage
S-curve lookup table bit width compression	bs	Confusion entropy, resistance to linear attacks	4×4-bit block compression mapping
Parallel path distribution structure	pm	Timing and side-channel attack resistance	Optimized memory access structure to reduce leakage risk
Intermediate state cache structure depth	cd	Power consumption and electromagnetic leakage window control	Dynamic buffer adjustment to avoid leakage hotspots

each communication interface, which is closely related to the selection of lightweight optimization strategies in the SM4 implementation. By incorporating these constraints into the system design, the proposed lightweight encryption mechanism can maintain stable operation across heterogeneous communication environments while avoiding excessive computational overhead. In order to visually illustrate the interaction of multiple parameters and how they affect security against resource constraints, Figure 2 illustrates the area distribution of security dimensions and resource utilization in a lightweight architecture.

In Figure 2, each area represents the security retention ability of a particular combination of policy combinations across different resource consumption dimensions. The horizontal axis represents the decrease rate of storage, and the vertical axis represents the entropy retention ratio. Larger areas indicate a more optimal synergy between safety and light weight design in this configuration. This diagram helps to select the best set of parameters based on the capability of the platform's resources during the system integration, and provides quantifiable support for the subsequent hardware adaptation of the communication security mechanism.

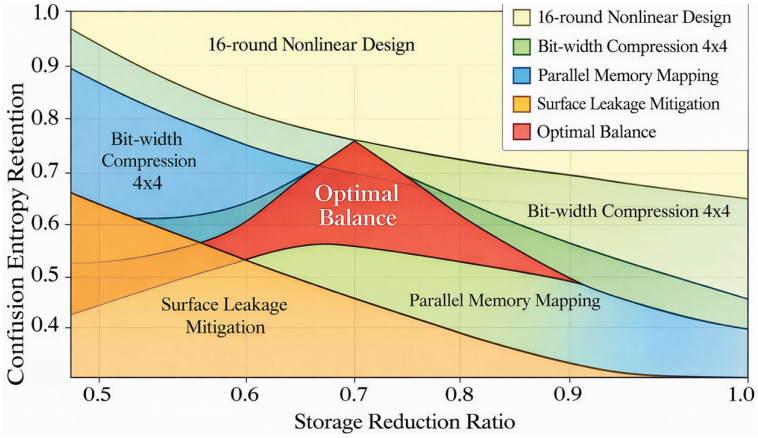

Figure 2 Security vs. resource constraint design area diagram.

4 Design and Implementation of Secure Communication Mechanisms for Communication Links Based on Lightweight SM4

4.1 Overall System Architecture Design

The model used in this study is a lightweight link-adaptive secure communication model for heterogeneous embedded links. The model takes plaintext frames, link-state parameters, session-key states, sequence numbers, and timestamp information as inputs, and generates ciphertext frames, integrity authentication values, and replay-decision outputs under constrained MCU resources. Structurally, the model consists of five tightly coupled components: a lightweight SM4 core, a key negotiation and update module, a link-state adaptation unit, an integrity verification and replay protection unit, and an embedded scheduling and resource-trimming unit. The model output is therefore not limited to encrypted data generation, but also includes secure transmission coordination and resource-aware deployability across UART, CAN, and BLE communication environments.

To satisfy the requirement of data communication under the circumstance of resource restriction and dynamic link, this paper puts forward a new structure based on light weight SM4. The overall architecture is shown in Figure 3. First, the encryption/decryption module uses Diffie-Hellman or SM2 Asymmetric Cryptography to set up an encryption channel between peers, and performs the key rotation through the periodic session key update mechanis. Second, the encryption/decryption module makes use of the optimized light

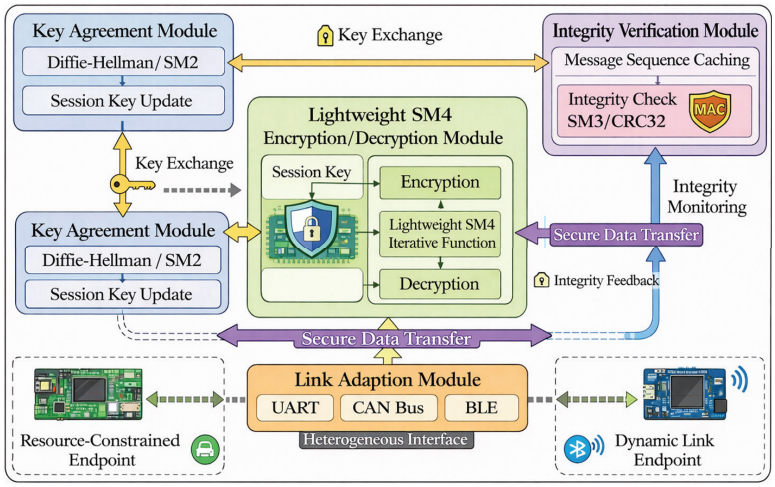

Figure 3 System architecture diagram of communication link security mechanism based on lightweight SM4.

weight SM4 round function, and integrates the link adaptive strategy to support the heterogeneous communication interface, including serial port, CAN bus and low-power Bluetooth. Third, based on SM3 or CRC32, the integrity verification module constructs Message Authentication Code (MAC), and includes a message sequence buffer and a timestamp window for replay attack detection. Finally, the system adheres to a modular, configurable design principle to accommodate variations in computation power, storage capacity, and interface types among the embedded MCU platforms, providing a secure security link with low resources and high security for edge deployment [19].

Figure 3 illustrates the overall architecture of the proposed lightweight SM4 secure communication framework. The diagram focuses on the system-level structure, including the interaction between the lightweight SM4 encryption module, the message integrity verification mechanism, and the heterogeneous communication interfaces such as UART, CAN, and BLE. Its purpose is to present the global deployment relationship among the security modules and the communication link components.

4.2 Key Negotiation and Key Update Mechanism

The Key Negotiation and Update Module is the basic safety pillar in the Light SM4 Driven Communication Link Security Mechanism. During initialization, the system generates a session master key through asymmetric

bootstrapping and maps it to an SM4 session key needed for symmetric encryption [16]. During the key-negotiation stage, the initial shared master key between the two communication parties is expressed as Equation (6):

$$K_0 = H((g^a \bmod p)^b \bmod p \| ID_A \| ID_B \| N_0) \tag{6}$$

where g is the public generator, p is the large prime modulus, a, b are the private random numbers for nodes A, B respectively, ID_A, ID_B represents the communication parties' identity identifiers, N_0 is the random perturbation factor introduced during the negotiation phase, and $H(\cdot)$ is the secure hash function used to eliminate structural dependencies and generate a fixed-length master key. Based on the negotiated master-key state, the dynamic session-key update rule used in this study is further defined in Equation (7):

$$K_{i+1} = H(K_i \oplus F(C_i, T_i)) \tag{7}$$

Where K_i denotes the session key for the ith round, C_i represents the current communication counter or message sequence state, T_i is the local time window parameter, and $F(\cdot)$ is a lightweight state mixing function that incorporates link dynamic characteristics. This design tightly correlates key evolution with communication behavior, allowing periodic key rotation without introducing additional interaction overhead. It provides an effective key input base for SM4 encryption/decryption and integrity authentication module [17].

4.3 Encryption/Decryption Process and Link Adaptation Mechanism Design

The SM4 encryption/decryption module constructs a data stream processing path based on circular function compression, register mapping, and interrupt-driven mechanism. This is combined with the state of the communication link to achieve a channel specific adaptive configuration [22]. Let each frame input plaintext data block be M_i. After key establishment, the round-execution structure of the lightweight SM4 encryption/decryption process under link-state adaptation is formulated as Equation (8):

$$C_i = E_K(M_i) = F^{32}(M_i, K_s) \oplus T_i \oplus \Phi(L_i) \tag{8}$$

Here, C_i denotes the encrypted ciphertext block for round i, K_s represents the current round key, $F^{32}(\cdot)$ is the lightweight round function iterator, T_i is the time synchronization vector, and $\Phi(L_i)$ is the link-layer parameter

mapping function. This function maps the correction vector of the buffer delay, the frame structure, and the interface register according to the type of link (for example, UART, CAN, BLE). To support cross-link adaptation and decryption synchronization, the system uses the state cached pipeline logic in the receiving side decryption process. Pseudocode is shown in Algorithm 2:

Algorithm 2 Lightweight SM4 decryption process based on link state mapping

```
void SM4_Link_Adaptive_Decrypt(FrameBuffer buffer, LinkState L_i, Key K_s) {
    for (int i = 0; i < buffer.frame_count; i++) {
        CipherBlock C_i = buffer.frames[i];
        // Step 1: Link Layer Adjustment Factor Calculation
        LinkAdjust adj = Compute_Link_Adjustment(L_i);
        // Step 2: Ciphertext adjustment (accounting for link timing offset)
        CipherBlock C_adj = XOR(C_i, adj.delta_vector);
        // Step 3: Execute SM4 Decryption Round Function
        PlainBlock M_i = SM4_Decrypt_Core(C_adj, K_s);
        // Step 4: Data Output
        OutputFrame(M_i);
    }
}
```

During runtime, the model operates in a closed loop. The negotiated session key initializes the lightweight SM4 core, the link-state mapping function adjusts buffering and timing-related parameters according to interface characteristics, and the integrity-replay unit verifies ciphertext legitimacy before data release. This closed-loop interaction allows the model to preserve both communication security and execution stability under dynamic link conditions.

4.4 Message Integrity Check and Replay Attack Protection Module

The Message Integrity Authentication and Replay Attack Protection Module serves as a parallel security control unit in the encryption/decryption process within the light weight SM4 encryption link. It is embedded in the data parsing path on the link's receiving side, with its overall structure depicted in Figure 4. For each received frame, the message authentication value generated by the hybrid checksum mechanism is written as Equation (9):

$$\mathrm{MAC}_i = H(C_i \| K_s \| N_i \| \Psi(L_i)) \tag{9}$$

where MAC_i represents the integrity check value for frame i, C_i denotes the encrypted ciphertext data block, K_s is the current session SM4 key, N_i is the incrementing message sequence number, $\Psi(L_i)$ is the link disturbance vector mapped from link state parameters, and $H(\cdot)$ denotes a lightweight hash or hybrid checksum function. The dual-constraint condition for replay detection is then given in Equation (10):

$$\Omega_i = \begin{cases} 1, & N_i > N_{\text{last}} \wedge |T_i - T_{\text{ref}}| < \Delta T \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where N_{last} denotes the maximum accepted sequence number, T_i represents the timestamp carried by the current packet, T_{ref} is the local reference time, and ΔT is the permitted time window threshold. By combining integrity verification with replay judgment, the final message-legitimacy discrimination function is defined in Equation (11):

$$V_i = \Omega_i \cdot \exp\left(-\frac{\|\mathrm{MAC}_i - \overline{\mathrm{MAC}}_i\|_2}{\eta}\right) \tag{11}$$

where $\overline{\mathrm{MAC}}_i$ is the locally recalculated authentication value, $\|\cdot\|_2$ denotes the L2 norm difference metric, and η is the security sensitivity adjustment coefficient. This design allows for both integrity verification and replay detection within a single decision-making framework, which can be adapted to a variety of link environments by parameterization. The module is loosely coupled to the SM4 decryption core, which ensures real time data processing while avoiding extra resources in an embedded system. This provides a stable interface for tailoring and deployment of embedded security mechanisms [19]. The control goals and computing resources of key variables are shown in Table 3.

Figure 4 describes the internal execution flow of the message integrity and replay protection mechanism. Instead of presenting the entire system architecture, this figure focuses on the operational procedure within the protection module, including ciphertext input, MAC recalculation, sequence number validation, timestamp comparison, and the final legitimacy decision.

Table 3 defines the key variables used in the message integrity and replay protection module and clarifies their security roles and storage overhead. Among them, the ciphertext block Ci is the protected message object, while the session key Ks controls the legitimacy of MAC generation. The sequence number Ni and timestamp Ti jointly support replay detection by combining monotonic packet identification with a transmission time window. The disturbance vector Ψ is introduced to encode link-state variations and strengthen

Table 3 Key variable descriptions and resource characteristics for message integrity and replay protection module

Parameter Name	Symbol	Security Control Function	Data Type	Resource Overhead (Bytes)	Description
Ciphertext Block	Ci	Input Data Integrity Hash Source	uint128_t	16	Encrypted Message Data Frame
Session Key	Ks	Control MAC Generation Legitimacy	uint128_t	16	Current Valid SM4 Symmetric Key
Message Sequence Number	Ni	Replay attack sequence detection	uint32_t	4	Unique packet identifier, monotonically increasing
Link Disturbance Vector	Ψ	Enhanced link-related entropy strength	uint32_t	4	Link-State-Encoded Disturbance Factor
Timestamp	Ti	Time window determination basis	uint32_t	4	Message transmission time, microsecond resolution
Locally computed MAC	$\overline{\mathrm{MAC}}_i$	Used for MAC difference comparison	uint128_t	16	Authentication value for locally recalculated ciphertext blocks
Discrimination function output	Vi	Final Message Legitimacy Judgment Scalar	float	4	Value range (0,1), used to control whether to accept the message

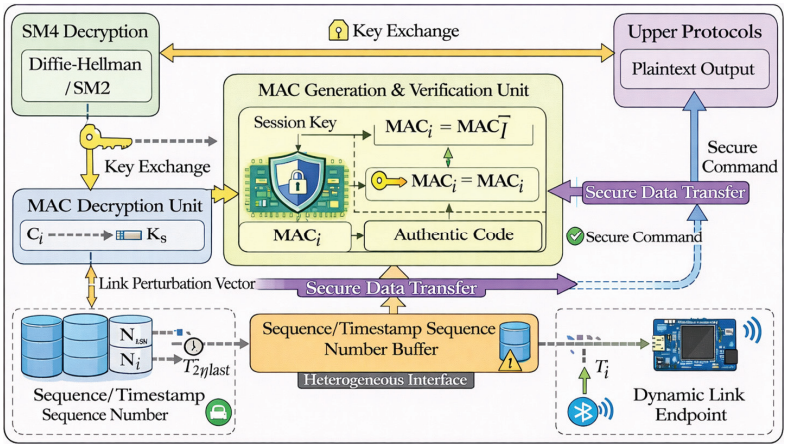

Figure 4 Module architecture for message integrity verification and replay attack protection in communication links.

entropy input under unstable communication conditions. The locally computed MAC is used for ciphertext authentication consistency checking, and the discrimination output Vi provides a scalar decision basis for final message acceptance. These variables together show that the module does not rely on a single authentication field, but forms a joint judgment mechanism integrating integrity verification, timing control, and link-state awareness.

4.5 Adaptation Strategy for Security Mechanisms in Embedded Environments

In order to efficiently operate communication link security mechanism on resource constrained embedded platform, this paper introduces module pruning, key cache repping, interrupt-driven coordination and so on. This ensures that the embedded platform's computing, power consumption, and storage limits are met, while preserving the core encryption power [20, 21]. To evaluate the executable security-processing capacity on embedded MCUs, the module-side throughput model is established as Equation (12):

$$R_{\text{eff}} = \frac{\alpha \cdot C_{\text{clk}}}{\beta_1 \cdot S_{\text{flash}} + \beta_2 \cdot M_{\text{ram}} + \beta_3 \cdot L_{\text{int}}} \tag{12}$$

where R_{eff} denotes the per-cycle security processing throughput on the embedded side, C_{clk} represents the main frequency control parameter, $S_{\text{flash}}, M_{\text{ram}}$ denote the Flash and RAM space occupied by the security

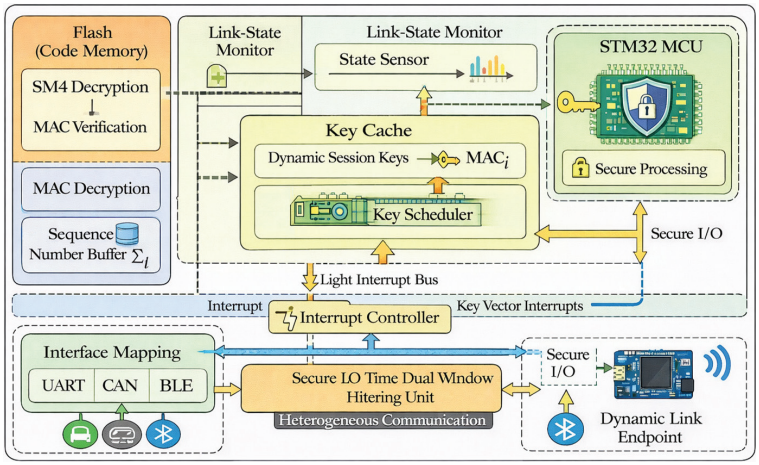

Figure 5 Module adaptation structure diagram of the lightweight SM4 communication security mechanism on embedded platforms.

module respectively, L_{int} indicates the interrupt service chain length, and a, β_i is the platform-related weighting coefficient. In order to further optimize the execution order and processing responsiveness of different security modules, the segmented scheduling objective is expressed as Equation (13):

$$F_{\text{opt}} = \min \left[\sum_{i=1}^{n} \left(\frac{t_i}{w_i} + \lambda_i \cdot \Delta P_i \right) \right] \tag{13}$$

where t_i denotes the processing time for the ith module, w_i represents its importance weight, ΔP_i indicates its power consumption fluctuation, and λ_i is the platform adjustment factor ensuring safety redundancy during multitasking switching under soft real-time conditions. Figure 5 shows this system's embedded deployment architecture on the STM32 family of MCUs, wherein the modules are coupled through a light-weight interrupt bus and are configured with dynamic key scheduling buffers and link-state-aware interfaces. Furthermore, in order to clarify deployment feasibility and resource matching between different microcontroller platforms, Table 4 lists computing performance, storage capacity, link adaptation and security module trimming strategies for typical embedded devices. Then, it provides the parameter reference and the selection of the module for the project implementation.

Table 4 summarizes the mapping relationship between hardware resources and security-module deployment strategies on typical embedded

Table 4 Module resource configuration and mapping relationships for typical embedded platforms

MCU Model	Clock Speed (MHz)	Flash Capacity (KB)	RAM Capacity (KB)	Deployable Modules	Adaptable Link Interface	Security Module Trimming Strategy
STM32F103	72	128	20	SM4 Decryption + MAC Verification	UART + CAN	SM4 Round Count Reduction + Table Compression
STM32L432	80	256	64	Full Module Deployment	UART + BLE	Full Reserve + Interrupt Optimization
GD32E230	64	64	8	SM4 Lite Edition	UART	Deploy only encryption + time window protection
ESP32-S2	240	1024	320	Full-featured + Heterogeneous adaptation	UART + WiFi/BLE	Multithreading + Dynamic Key Scheduling

platforms. The differences in clock speed, Flash capacity, and RAM capacity directly determine whether the platform can support full deployment or only a trimmed version of the proposed mechanism. For example, STM32L432 and ESP32-S2 can support full-module deployment because of their relatively sufficient storage and runtime resources, whereas GD32E230 is limited to the lightweight edition with only encryption and time-window protection retained. The adaptable link interface column further indicates that module deployment is not only constrained by hardware capacity, but also by communication-interface characteristics. Therefore, the trimming strategies listed in the last column reflect the platform-oriented implementation principle of the proposed method, namely preserving core security functions while matching the actual resource boundary of each device.

5 Experimental Platform Setup and Performance Evaluation

5.1 Experimental Platform and Test Environment Configuration

In order to verify the deployment and operation efficiency of the lightweight SM4 Driver Communication Link Security Mechanism, a hardware and

software cooperative experiment platform was built. The platform uses STM32L432KC (80 MHz, 64 KB RAM, 256 KB Flash) as core microcontroller. It includes custom link-adaptation modules (UART, CAN, BLE) and is coupled to the Ubuntu 22.04 host system through an external debugger to interact with the real time log capture. The experimental dataset includes: (1) 3,600 link-type coverage test entries covering the full transmission cycle of the three communication interfaces; (2) 1,200 entries documenting the key negotiation and session update processes to verify the stability of the key state transition; (3) 2,000 encryption/decryption and integrity check operations, all taken from the actual transmission log, to ensure that the test scenario covers typical boundary conditions, such as link dynamic jitter, packet loss retransmission, and latency skew. The platform supports high precision timing and link anomaly simulation, providing a realistic and controllable implementation environment for later performance testing [26].

5.2 Encryption/Decryption Latency and Throughput Testing

Based on the STM32L432KC embedded platform, encryption/decryption latency and throughput of the lightweight SM4 module were evaluated across UART, CAN, and BLE interfaces [23], Test frames contained 128-byte payloads. Execution time, interface buffer latency, and data processing capacity were recorded. After excluding the top and bottom 5% outliers, mean values were calculated. Key performance results are summarized in Table 5.

Analysis of the test results in Table 5 shows that the BLE interface provides the best performance. The average encryption/decryption delay is 165.8 μs and 170.2 μs, which is much lower than the 207.5 μs and 213.1 μs of the CAN. This shows that it has a high real time response capability in high frequency data exchange scenarios. Additionally, BLE achieves the maximum throughput of 8.42 KB/s, which shows that there is more optimization potential in the area of link bandwidth utilization and cache consistency. UART has demonstrated stable latency control with a maximum deviation

Table 5 Encryption/decryption latency and data throughput performance test results across different link interfaces

Communication Interface	Average Encryption Latency (μs)	Average Decryption Latency (μs)	Maximum Latency Deviation (μs)	Throughput (KB/s)
UART (115200 bps)	182.4	188.9	$\pm$13.7	6.71
CAN (500 kbps)	207.5	213.1	$\pm$15.4	5.26
BLE (1 Mbps)	165.8	170.2	$\pm$11.2	8.42

of only ± 13.7 μs, which is ideal for applications that require high timing consistency; it has a throughput of 6.71 kB/s, which is moderated. The CAN interface has the highest latency with noticeable fluctuations (± 15.4 μs), but it has demonstrated excellent anti-interference and protocol stability in an industrial bus system, which makes it suitable for controlling communication in a complicated electromagnetic environment. All in all, the lightweight SM4 provides stable delay control and acceptable throughput for all three interfaces, proving its suitability and practicability for embedded deployment across heterogeneous links [25].

5.3 Performance Comparison with Original AES/SM4 Versions

To verify the feasibility of SM4 in embedded communication security, three schemes – AES-128, SM4, and lightweight SM4 – were implemented under identical compilation settings, link protocols, and frame structures on the STM32L432KC platform. Tests were conducted over BLE with 100 transmission cycles per group, measuring encryption latency and throughput. The performance distribution is presented in Figure 6.

As shown in Figure 6, lightweight SM4 exhibits distinct clustering in both encryption latency and throughput. Its data points predominantly cluster within the latency range of 160–180 μs and throughput range of 8.0–8.6 KB/s, demonstrating excellent computational efficiency and

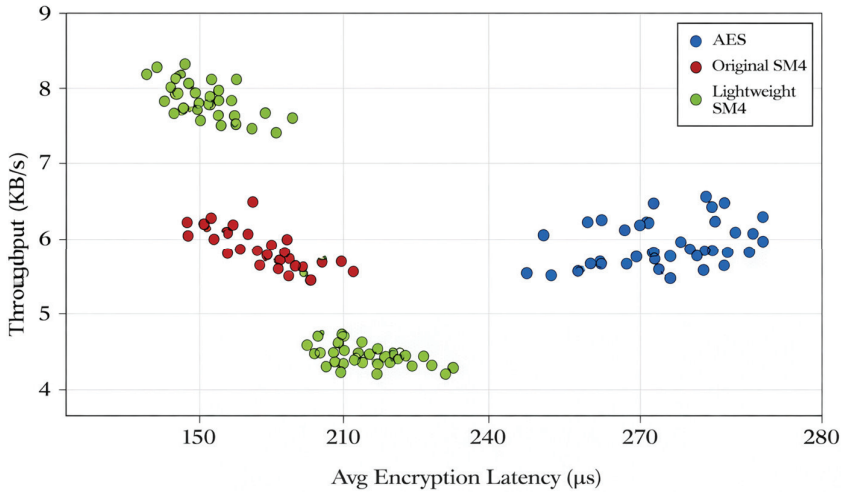

Figure 6 Performance scatter plot of AES, Raw SM4, and lightweight SM4 algorithms on BLE link.

transmission coordination on low-power platforms. The raw SM4 exhibits a broader scatter range, with latency fluctuations between 210–240 μs and throughput dropping to 5.5–6.3 KB/s. This reflects the fact that its architecture is not optimized for embedded systems, leading to longer memory access paths. AES shows the worst performance, with scattered points showing high latency and low throughput, primarily clustered in the 250–270 μs and 4.8–5.5 KB/s ranges. Overall, the lightweight SM4 exhibits high density, low dispersion properties under BLE links, which confirms its architecture compatibility with the embedded platform and its capability to guarantee real-time communication. This provides the basis of performance for later large-scale deployment.

5.4 Security Validation and Anti-Attack Capability Assessment

To evaluate SM4 security on resource-constrained platforms, multi-dimensional attack simulations and disturbance tests were conducted under BLE communication, including differential attacks, replay attempts, power leakage sampling, and electromagnetic interference. Tests on the STM32L432KC platform measured decryption success rate, authentication pass rate, and state recovery delay. Security performance is summarized in Table 6, and fault-tolerant response delay is shown in Table 7.

As shown in Table 6, the system has proven to be robust to a variety of typical attacks. The success rate of decryption in replay attack is 100%, which shows that the CDN is able to block multiple packets with stable response. However, the MAC authentication pass rate is slightly lower (93.1%), indicating that the higher attack rate may result in a partial authentication mapping delay and mismatch. The success rate of the differential attack and the voltage

Table 6 System security resilience statistics under typical attack scenarios

Attack Type	Attack Severity Level	Decryption Success Rate (%)	MAC Verification Pass Rate (%)	State Recovery Time (ms)
Differential Attack	Moderate	98.7	97.3	2.3
Replay Attack	High	100	93.1	1.7
Side-Channel Energy Sampling Interference	Medium	96.4	94.6	2.9
Electromagnetic Interference Simulation	Low	99.5	98.2	2.1
Voltage Dynamic Fluctuation Disturbance	Moderate	97.1	95.5	2.6

Table 7 Fault-tolerant response performance of the security module under error injection conditions

Injection Type	Injection Frequency (Hz)	Message Error Rate (%)	Automatic Retransmission Trigger Rate (%)	System Stable Recovery Period (ms)
Timestamp Perturbation	10	2.1	3.4	4.6
Session Key Offset	5	1.8	2.7	3.9
Packet Sequence Misalignment	20	2.9	4.1	5.2
Random code injection interference	15	3.2	4.6	5.5

disturbance scenario is 98.7% and 97.1%, and the recovery time is 2. 3 ms and 6 ms respectively. This shows that the system is capable of performing fast rollback and context validation even in moderate disturbances. MAC validation pass rate has fallen to 94.6% under side-channel power sampling interference, suggesting a slight disruption of the S-box processing path, but overall performance remains within acceptable safety limits. Overall, this architecture demonstrates engineering-grade adaptability to multiple source attacks while controlling resource consumption.

Analysis of the error injection data in Table 7 shows that the system has demonstrated robust fault tolerance and self-repairing capabilities over a variety of disruption scenarios. In particular, in the time stamp interference test at 10 Hz, the error rate of the system packet is 2.1%, the auto-retransmission trigger rate is 3.4%, and the recovery period is only 4.6 ms. This shows that the system has a fast detection and response mechanism for small variations in the transmission time. In the key offset scenario, the error rate was controlled at 1.8%, and the recovery period was shortened to 3.9 ms, demonstrating that the state consistency verification mechanism was effective in ensuring accuracy and stability in key matchs.essage sequence misalignment and random injection resulted in more significant data structure disruption, with error rates increasing to 2.9% and 3.2%, respectively. The rate of retransmissions was 4.1% and 4.6%, and the recovery period was slightly prolonged to 5.2 ms and 5.5 ms. Communication disruption has not yet been resolved. Overall analysis of all data shows that the system achieves closed loop correction in the event of a burst disturbance. This enhances the overall mechanism's practical stability in dynamic scenarios such as Internet of Things and automotive applications.

5.5 Feasibility Analysis for Deployment in Typical Communication Links

The lightweight SM4 security mechanism was deployed in embedded communication links using UART, CAN, and BLE interfaces under a consistent MCU architecture. Without DMA or advanced interrupt scheduling, the system completed channel initialization, key synchronization, and interface configuration. Deployment latency and processing load were measured during real communication cycles. Key deployment metrics across different links are summarized in Table 8.

Table 8 data shows that the deployment of the lightweight SM4 mechanism maintains controllable resource overhead in all three communication link configurations: Flash usage is maintained in the 18–20 KB range, and RAM consumption is no more than 9 KB, thus meeting the operating requirements of low to mid range microcontrollers. The UART interface shows excellent lightweight deployment features with minimum deployment delay (14.6 ms) and the simplest configuration complexity (2), so it can be used to deal with control oriented periodic tasks. The CAN link has a slightly higher cost of resources and configuration, but its structure stability and message arbitration make it more suitable for multi-node real-time systems. The BLE, because of the increased load of the protocol stack and the complexity of the interface, has a deployment delay of 21.5 ms and a maximum number of configuration commands (52), which needs to be adjusted by platform scheduling capabilities. Comprehensive analysis shows that the lightweight SM4 solution, in multi-link heterogeneous environments, allows customized deployment to be tailored to the interface features through a modular configuration, effectively balancing the integrity of the security mechanism with the constraints of embedded resources.

Table 8 Comparison of deployment performance and resource overhead across communication links

Communication Link	Initial Deployment Latency (ms)	Flash Memory Occupied by Security Module (KB)	RAM Usage (KB)	Interface Compatibility Complexity (1–5)	Number of Configuration Commands (Items)
UART	14.6	18	7.2	2	34
CAN	17.3	20	8.1	3	41
BLE	21.5	19	8.7	4	52

6 Conclusion

This study developed a lightweight SM4-based communication security mechanism for heterogeneous embedded links, aiming to achieve secure transmission under resource-constrained conditions while preserving real-time responsiveness and deployment feasibility. By integrating S-box structure compression, round-function optimization, key-scheduling reconstruction, module trimming, and interface adaptation, the proposed method established a lightweight and configurable security framework for UART, CAN, and BLE communication environments. Experimental results on the STM32L432KC platform demonstrated that the proposed mechanism achieved an average encryption latency of 165.8 μs and a throughput of 8.42 KB/s under the BLE link, while maintaining stable delay control across UART and CAN interfaces. In addition, the deployment results showed that the security module required only 18–20 KB Flash and no more than 9 KB RAM across different link configurations, indicating favorable embedded deployability. Security validation further confirmed that the mechanism retained strong resilience under replay attacks, differential attacks, electromagnetic interference, and error-injection disturbances. These findings support the original assumption that coordinated optimization of SM4 lightweight implementation and link-level security mechanisms can simultaneously improve security, real-time performance, and engineering applicability on constrained embedded platforms. Future work should focus on three aspects: finer-grained interrupt scheduling and energy-aware adaptation for ultra-low-power devices, communication-state-aware dynamic key update under high-concurrency links, and more rigorous cryptographic security validation to evaluate structural equivalence and side-channel resistance of the lightweight SM4 architecture.

References

[1] Chen X, Li B, Zhou Q. Lightweight and High-Performance Data Protection for Edge Network Security[J]. Wireless Communications and Mobile Computing, 2022, 2022(1): 8458314.

[2] Goulart A, Chennamaneni A, Torre D, et al. On wide-area IoT networks, lightweight security and their applications—a practical review[J]. Electronics, 2022, 11(11): 1762.

[3] Justindhas Y, Jeyanthi P. Secured model for internet of things (IoT) to monitor smart field data with integrated real-time cloud using

lightweight cryptography[J]. IETE Journal of Research, 2023, 69(8): 5134–5147.

[4] Wang J. Identification of SQL Injection Security Vulnerabilities in Web applications Based on Binary Code Similarity. Journal of Cyber Security and Mobility, 2024, 13(6): 1239–1262. DOI: 10.13052/jcsm2245-1439 .1361.

[5] Zhang H, Meng F, Wang Q. Computer Network Security System Optimization Based on Improved Neural Network Algorithm and Data Search. Journal of Cyber Security and Mobility, 2025, 14(1): 75–100. DOI: 10.13052/jcsm2245-1439.1414.

[6] Kumar S, Kumar D, Dangi R, et al. A review of lightweight security and privacy for resource-constrained IoT devices[J]. Computers, Materials and Continua, 2024, 78(1): 31–63.

[7] El Hanine M, El-Yahyaoui A, Es-Sadaoui R. Design and assessment of lightweight cryptographic algorithms on ESP32 and STM32 for IoD security[J]. Egyptian Informatics Journal, 2025, 32: 100818.

[8] Le X, Li J, Zhao Y, et al. A Security-Enhanced Scheme for ModBus TCP Protocol Based on Lightweight Cryptographic Algorithm[J]. Electronics, 2025, 14(18): 3674.

[9] Wang Y, Xu Y, Liu Z, et al. Research on Lightweight Dynamic Security Protocol for Intelligent In-Vehicle CAN Bus[J]. Sensors, 2025, 25(11): 3380.

[10] Li X, Li M, Xu L, et al. Lightweight identity authentication and key agreement scheme for VANETs based on SSL-PUF[J]. Scientific Reports, 2025, 15(1): 21469.

[11] Chen Y, Ou W, Pang M, et al. A cross-chain model for warehouse receipts in port supply chain based on notary mechanism and ShangMi cryptographic algorithms[J]. Scientific Reports, 2025, 15(1): 14390.

[12] Sevin A, Mohammed A A O. A survey on software implementation of lightweight block ciphers for IoT devices[J]. Journal of Ambient Intelligence and Humanized Computing, 2023, 14(3): 1801–1815.

[13] Islam N, Rahman M S, Mahmud I, et al. A blockchain-enabled distributed advanced metering infrastructure secure communication (BC-AMI)[J]. Applied Sciences, 2022, 12(14): 7274.

[14] Samanth S, KV P, Balachandra M. Security in internet of drones: A comprehensive review[J]. Cogent Engineering, 2022, 9(1): 2029080.

[15] Sutar S, Mekala P, Kameswari U S. Secured IoT node design through protected ITULES family group-II (ULWC) implementation against

physical attacks for ubiquitous computing[J]. International Journal of Embedded Systems, 2023, 16(4): 288–311.

[16] Zakaria A A, Ab Halim A H, Ridzuan F, et al. LAO-3D: A symmetric lightweight block cipher based on 3d permutation for mobile encryption application[J]. Symmetry, 2022, 14(10): 2042.

[17] Sim M, Eum S, Kwon H, et al. Optimized Implementation of Simpira on Microcontrollers for Secure Massive Learning[J]. Symmetry, 2022, 14(11): 2377.

[18] Papaioannou C, Dimara A, Papaioannou A, et al. Hierarchical Resources Management System for Internet of Things-Enabled Smart Cities[J]. Sensors, 2025, 25(3): 616.

[19] Khan N A, Jhanjhi N Z, Brohi S N, et al. A secure communication protocol for unmanned aerial vehicles[J]. CMC-Computers Materials & Continua, 2022, 70(1): 601–618.

[20] Sun Q, Lin K, Si C, et al. A secure and anonymous communicate scheme over the internet of things[J]. ACM Transactions on Sensor Networks (TOSN), 2022, 18(3): 1–21.

[21] Du H, Wang J, Niyato D, et al. Rethinking wireless communication security in semantic Internet of Things[J]. IEEE Wireless Communications, 2023, 30(3): 36–43.

[22] Bondada P, Samanta D, Kaur M, et al. Data security-based routing in MANETs using key management mechanism[J]. Applied Sciences, 2022, 12(3): 1041.

[23] Illi E, Qaraqe M, Althunibat S, et al. Physical layer security for authentication, confidentiality, and malicious node detection: A paradigm shift in securing IoT networks[J]. IEEE Communications Surveys & Tutorials, 2023, 26(1): 347–388.

[24] Ying Z, Wang K, Xiong J, et al. A literature review on V2X communications security: Foundation, solutions, status, and future[J]. IET Communications, 2024, 18(20): 1683–1715.

[25] Cheng T, Wu Z, Wang C, et al. Research on vehicle-to-cloud communication based on lightweight authentication and extended quantum key distribution[J]. IEEE Transactions on Vehicular Technology, 2024, 73(8): 12082–12095.

[26] Chai S, Yin H, Xing B, et al. Provably secure and lightweight authentication key agreement scheme for smart meters[J]. IEEE Transactions on Smart Grid, 2023, 14(5): 3816–3827.

Biography

Xiaoli Tang obtained her Master of Engineering in Control Theory and Control Engineering (2009) from NJUST, Nanjing. Presently, she is working as a Lecturer in the School of Electronic Engineering and Optoelectronic Technology, Nanjing University of Science and Technology ZiJin College, Nanjing. She previously worked as a Software Development Engineer at a leading Chinese telecommunications company for 15 years. She has published 2 papers in Chinese journals and holds 1 invention patent. She is currently a Senior Engineer. Her areas of interest include computer network communication, network access communication protocol and network security.