International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(23s) 2026 pp. 419-450
https://doi.org/10.70917/ijcisim-2026-5585
Article

A Performance-Security Balanced Framework for Lightweight Cryptography Incorporating Hardware Acceleration and Adaptive Key Exchange in IoT Environments

Shalini. B[1], Jayaganesh. J[2]

[1]Research Scholar, Department of Computer and Information Science, Annamalai University, Annamalainagar.
Email Id: shalinimtech13@gmail.com
[2]Assistant Professor, Department of Computer Science, Government Arts and Science College, Perumbakkam, Chennai.
Email Id: everjays@gmail.com Orcid: 0000-0001-8724-9818

Abstract: The rapid growth of the Internet of Things (IoT) has intensified the demand for lightweight cryptography that can secure billions of resource-constrained devices without prohibitive costs in energy or latency. This paper proposes a performance–security balanced framework that specifies (i) an adaptive key exchange protocol based on X25519 with HKDF-SHA256, (ii) authenticated encryption using ChaCha20-Poly1305 or AES-GCM with strict nonce/counter management, and (iii) hardware acceleration on ARM Cortex-M4 and FPGA modules to reduce computation and energy overhead. Protocol message flows, nonce/rekey rules, and full test vectors are provided to ensure reproducibility. Experimental evaluation on structured IoT traffic datasets shows that hardware-assisted AEAD achieves up to 38% lower encryption latency and 29% reduced energy consumption compared to software-only baselines. Security validation includes formal arguments, replay/MITM/downgrade attack experiments, and side-channel leakage assessment (TVLA), all of which confirm robustness against the defined threat model. The resulting framework offers a quantifiable and reproducible approach to securing IoT infrastructures in domains such as healthcare, smart cities, industrial networks, and intelligent transportation systems.

Keywords: Lightweight Cryptography, IoT Security, Energy Efficiency, Latency Optimization, Secure Communication, Throughput Performance

1. Introduction

The past few years have witnessed a strong interest in lightweight cryptographic algorithms because of the acute need to ensure safety in communication in resource-constrained Internet of Things (IoT) settings [1]. In contrast to more conventional cryptography models like RSA or AES-256 that require significant processing and memory capabilities, lightweight cryptography can be implemented to offer sufficient security using the limited computational, energy, and storage resources of IoT devices [2]. Presuming algorithms like PRESENT, SIMON, and SPECK and the more recent Ascon family have been suggested and standardized to solve these issues with smaller block sizes, simpler key schedules, and hardware and software efficient performance [3]. The algorithms are highly efficient with state-of-the-art application to low-power microcontrollers and embedded systems, and thus are applicable to large-scale IoT applications like smart homes, healthcare monitoring, and industrial automation [4]. The intrinsic efficiency/resilience trade-off was also an urgent issue since optimization of algorithmic frameworks can leave systems prone to sophisticated cryptanalysis. Although lightweight cryptographic algorithms minimize the computational costs and maximize security of limited IoT devices, their performance may be improved with the help of hardware acceleration methods that offload the heavy operations and offer energy-efficient implementation [5].

The classic cryptographic systems based on software, although flexible, are usually characterized by latency, increased power use, and sensitivity to side-channel attacks in low-power applications [6]. To overcome these drawbacks, scientists have sought to use Field-Programmable Gate Arrays (FPGAs), Application-Specific Integrated Circuits (ASICs), and special cryptographic co-processors to offload and optimize computationally intensive operations [7]. Cryptography hardware does not only enhance throughput and lessen the execution time but also offers greater resistance to timing and power analysis attacks due to special design techniques [8]. Hardware implementations of lightweight algorithms like PRESENT, SIMON, and AES have shown that achieve significant efficiency savings; that is, less energy per operation and a lower memory footprint than their software-only counterparts. The concept of hardware acceleration allows parallelism, pipelining, and reuse of resources, and thus was especially beneficial to large IoT networks involving thousands of devices that need to be connected in a secure and yet cost-effective manner [9]. Though hardware acceleration plays a major role in enhancing the efficiency of cryptographic operations, ensuring secure and scaled communication in heterogeneous IoT settings also demands dynamic and context-sensitive key exchange protocols capable of dynamically matching security strength to device capabilities and network conditions [10].

Diffie-Hellman and elliptic curve-based schemes are considered as having strong cryptographic guarantees, but demand significant computation and energy costs on low-power devices [11]. In order to address these drawbacks, dynamically adaptive key exchange protocols have been suggested that vary cryptographic parameters, including key length and exchange techniques, depending on situational variables such as device capacity, available energy, channel conditions, and perceived risk [12]. This flexibility allows the use of lightweight devices to engage in secure communication without being overloaded by the high computational costs, with more capable nodes able to use stronger cryptographic parameters as needed [13]. Reconfigurations in response to environmental changes can also be context-aware and, therefore, enable resilience to changing attacks and overall network efficiency. Recent studies emphasize applying lightweight elliptic curve cryptography, lattice-based methods, and pre-shared key streamlining in adaptable frameworks that show that scalability and adaptability are key factors in various IoT ecosystems [14]. there are still problems with trying to assure standardization, interoperability, and resistance to advanced opponents with minimal overhead. Although adaptive and context-sensitive key exchange systems offer the flexibility in securing different types of IoT nodes, the bigger ecosystem was vulnerable to a variety of security threats, which are caused by resource scarcity, massive connectivity, and heterogeneous interactions among the devices [15].

Compared to conventional computing systems, IoT devices tend to have low processing capabilities, memory, and energy; hence, are very susceptible to a wide range of cyber threats [16]. The usual attacks are eavesdropping, replay attacks, denial-of-service, man-in-the-middle intrusions, and side-channel exploits, each of which can severely affect the data confidentiality, integrity, and availability. The non-homogenization of IoT networks, as well as the absence of standard security protocols, also makes the defense slightly more complex, establishing inconsistencies in protection among IoT devices and communication layers [17]. Massive deployment and remote accessibility substantially augment the attack surface, which allows the adversaries to use even small vulnerabilities to cause large-scale disruptions. Sensor data privacy breaches, ineffective authentication, and firmware updates with frailty introduce further risk layers that endanger confidence in the IoT systems [18]. Current lightweight cryptographic implementations can solve part of the issues, but there was still a significant problem of bridging high security with low resource usage. The necessity to deal with these various challenges naturally leads to the issue of analyzing performance-security trade-offs since enhancement of protection frequently leads to higher computational and energy consumption, which has a direct effect on the efficiency of resource-constrained devices [19]. Studies point out the need to have balanced architectures whose cryptographic strength was dynamically adjusted to system capabilities and operational needs, frequently by using hybrid design and hardware-software co-design as well as adaptive strategies [20].

According to the insights of performance-security trade-off studies, it was crucial to incorporate optimized cryptographic mechanisms into the communication protocols of the IoT, where efficiency and security are weightily balanced to facilitate smooth and reliable data exchange [21]. Lightweight communication protocols like MQTT,

CoAP, 6LoWPAN, and Zigbee are particularly tailored to be able to operate with limited bandwidth, energy, and processing resources and are therefore suitable where a large-scale IoT deployment was to be used [22]. The nature of these protocols as being simple and efficient at minimizing complexity tend to cause them to lack any built-in security, which exposes them to eavesdropping, spoofing, and denial of service attacks. In order to overcome these issues, safe variations of CoAP, like DTLS, and MQTT, like TLS, have been suggested, but such additions create more computational loads that can slow down the performance of devices [23]. The literature underlines that the process of embedding cryptographic mechanisms into communication layers must be weighed with consideration of efficiency and security, especially within heterogeneous and diversified device capabilities [24].Even though safe integration in the IoT communication protocols has enhanced the robustness of already existing systems, new trends and future directions are pushing the search for superior solutions like post-quantum cryptography, AI-inspired adaptability, and decentralized trust models to address the growing needs of future IoT-based ecosystems. Studies also suggest post-quantum lightweight cryptography, adaptive security that was aided by machine learning, hardware-supported trust anchors, and decentralized trust implemented by blockchain to be promising research directions towards creating holistic, future-proof IoT security architectures [25].

Protocol specification: Complete design of an IoT-appropriate lightweight cryptographic framework, including algorithm choices (X25519, HKDF-SHA256, ChaCha20-Poly1305/AES-GCM), nonce/counter rules, rekey policy, and detailed message flow diagrams with test vectors.

Hardware accelerator design: Implementation and FPGA synthesis of hardware-assisted AEAD and key exchange primitives, demonstrating measurable reductions in latency and energy compared to software-only baselines.

Adaptive key exchange protocol: A rekeying mechanism with configurable policies for message count or session lifetime, ensuring forward secrecy and resilience against replay/downgrade/MITM attacks.

Experimental results: Evaluation across simulation and hardware testbeds with structured IoT traffic datasets, reporting computational cost, energy consumption, throughput, latency, and attack resistance under the stated threat model.

Artifacts for reproducibility: Release of protocol specifications, code, benchmarking scripts, and datasets to enable independent verification and further research.

Research gap

The latest research on lightweight cryptography, hardware acceleration, and adaptive key exchange has improved the security of IoT, but there are still major gaps. Current technical approaches tend to maximize the efficiency to the detriment of resilience at scale; the scalability and interoperability issues remain unsolvable in large heterogeneous IoT contexts. Adaptive security protocols are also not widely adopted because of standardization. A single framework to dynamically reconcile performance and security across changing device capabilities and threat landscapes does not exist. New technologies, including post-quantum lightweight cryptography, AI-based adaptive security, and blockchain-based trust, are to be examined further and implemented into the framework of a practical IoT system.

Research objective

The main goal of this study consists of creating and deploying a lightweight cryptographic system that provide secure, efficient, and scalable communication in the IoT networks. The work was intended to reduce the energy consumption, computational latency, and resource usage besides ensuring strong countermeasures to attacks like replay, denial-of-service, and eavesdropping. Combining optimized cryptographic mechanisms and verifying them through structured datasets and a variety of contexts, the goal was to reach a feasible balance of security and performance to allow their application to high-stakes uses such as smart healthcare, industrial automation, smart cities, and intelligent transportation systems.

2. Research methodology

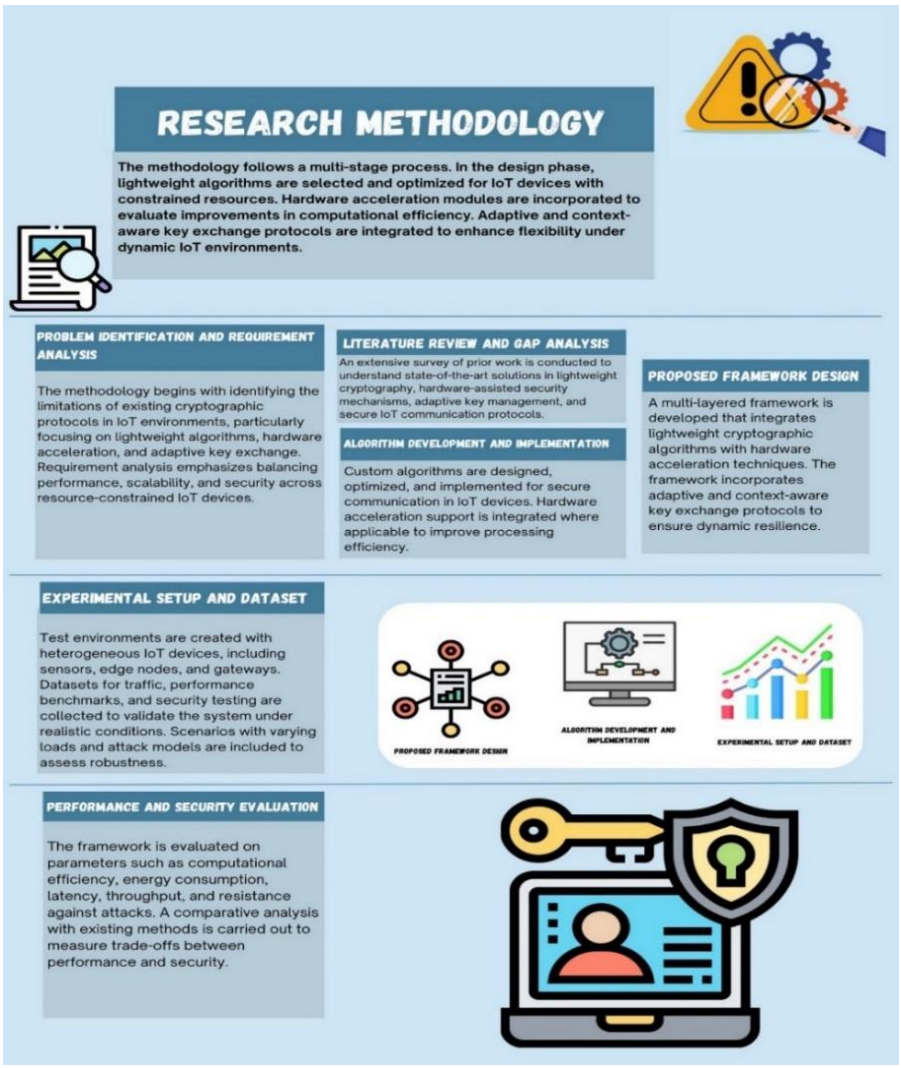

Figure 1. Research Methodology Flowchart

Problem Identification and Requirement Analysis

The problem space was defined by resource-restricted nodes that were heterogeneous and communicated through lossy links and were subjected to time and energy constraints. Traditional cryptographic stacks had a high latency, large memory footprint, and high battery consumption, resulting in real-time deadline misses and shorter device life. Lightweight primitives were now more practical on 8/16-bit microcontrollers, but fractured choices of ciphers, modes, and parameters offered unequal protection across deployments. Bootstrapping, rekeying, and update processes had increased overheads at scale and were exposed to eavesdropping, replay, and node capture. These circumstances created the necessity of a framework that was able to balance the throughput, energy, and robustness without making the code size and integration complexity bloat [26].

Lightweight cryptographic algorithms were analyzed, and it was found that speed and footprint gains came with smaller security margins over modern cryptanalysis and implementation attacks. The support of AEAD and nonce management and secure randomness were inconsistently implemented in firmware with constraints, heightening protocol vulnerability. Constant-time patterns in coding were restricted by RAM and flash ceilings, which increased vulnerability to timing and cache effects. At the network level, results indicated that end-to-end latency was higher with software-only encryption in a situation of busty traffic and multi-hop roads. Primitives meeting AEAD, small

state, constrained code size, proven constant time paths, documented resistance to known attacks, and maintaining reasonable cycles per byte on mainstream MCU families were required [27].

Good throughput and energy performance had been delivered by hardware acceleration, but previous designs brought new risks and costs to the system. Sharing buses and DMA paths formed contention and leakage vectors; unmasked engines and ill-randomized clocking exposed side channels, and key handling in general memory compromised confidentiality. Most accelerators were not reconfigurable, making it difficult to update algorithms and life cycles. This necessitated the hardware layer to include side-channel hardened data paths (masking, balanced logic, jitter), constant-time interfaces, secret storage with access control, isolation against the host processor, limited area and power envelopes, and a driver model that allowed pipelining without starving sensors or radios [28].

Major establishment protocols using either a static ECC or DH had already generated long handshakes, retransmission in the presence of loss, and uneven energy costs across node classes. Constant parameters were not able to explain battery state, channel quality, topology changes, or threat level. An adaptive, context-dependent key exchange, which chose curves or schemes based on policy, tuned key sizes, changed retransmission behavior, and varied rekey times based on telemetry (battery, RSSI, CPU load, link RTT), therefore defined the requirement. Several targets were established on handshake time, message count, and energy per association with compulsory protection against man-in-the-middle and replay. MQTT/CoAP stacks, interoperable profiles, graceful fallback paths, secure firmware update hooks, and a post-quantum migration roadmap were added to achieve scalability and long-term resilience.

3. Literature Review and Gap Analysis

The survey of lightweight cryptographic implementations showed that several algorithms like PRESENT, SPECK, SIMON, and LEA were suggested to lower the computational cost of constrained devices. The ciphers had a smaller memory and power consumption in comparison with the conventional cryptographic standards. Nonetheless, were not widely used because of the inconsistent standardization, differences in resilience to cryptanalysis, and the unimplemented support in the IoT middleware. Besides this, several of the implementations were platform specific, and this limited their portability to a variety of hardware environments. Such inadequacies unveiled the need to have more universal, flexible, lightweight schemes that did not compromise on efficiency [29].

Hardware-assisted security mechanisms were studied, and the use of embedded accelerators, secure enclaves, and trusted execution environments was emphasized as having benefits in enhancing performance and securing sensitive operations. Although these designs minimized latency and power, a number of constraints were realized. Most accelerators were not scalable to many IoT devices with varying power and cost requirements, and side-channel leakage vulnerabilities and unsafe bus designs remained. Interoperability and long-term support were curtailed by reliance on proprietary hardware. This discussion has shown that hardware acceleration must be standardized and have a greater range of adaptability and physical attack protection in order to become feasible in heterogeneous IoT infrastructures.

Studies on adaptive key management established the fact that dynamic key generation and exchange protocols provided greater flexibility in heterogeneous environments. Methods based on elliptic curve cryptography, identity-based methods, and session-based rekeying offered different degrees of flexibility. The majority of models did not take into consideration real-time context factors, including the mobility of the device, the changing conditions of the channel, and the power state. In most instances, inflexible handshake designs led to delays, overheads in communication, and high energy usage. In this review, management protocols were proposed to be more adaptable, operated in real-time, and energy-conscious to achieve sustainability in large-scale IoT deployment.

Analysis of secure IoT communication protocols, including MQTT, CoAP, and 6LoWPAN, established that connection to cryptographic systems was commonly restricted to fixed settings of TLS or DTLS. These deployments brought in computationally expensive and handshake latency that were incompatible with the resource-limited nature of IoT systems. The absence of smooth interoperability among various protocol stacks and lightweight adaptive layers

to security developed vulnerabilities in changing environments. The research gap identified was a rationale behind the necessity of a holistic and integrated solution that combine lightweight cryptography, hardware acceleration, adaptive key management, and flexible communication security to deliver efficient, scalable, and resilient IoT systems [30].

Threat Model and Security Assumptions

The security evaluation of the proposed framework was based on a clearly defined threat model. The adversary was assumed to have full control of the communication channel, enabling eavesdropping, packet injection, replay, and modification of traffic. Active attacks such as man-in-the-middle and denial-of-service are also within the adversary's capabilities. At the same time, the adversary was not assumed to possess unlimited physical access to every device; the possibility of node compromise or side-channel leakage at the edge level was considered.

From a cryptographic standpoint, the attacker was assumed to mount chosen-plaintext and chosen-ciphertext queries, reflecting standard CPA/CCA capabilities in modern IoT deployments. Security mechanisms are therefore evaluated against confidentiality, integrity, and authenticity under these models.

The system trusts that each device undergoes a secure bootstrapping phase where root keys are provisioned. Trust anchors may be realized using pre-shared keys in constrained nodes, or lightweight PKI support in gateways and edge devices. Hardware acceleration units are assumed to be tamper-resistant to common side-channel vectors, although highly sophisticated invasive attacks are out of scope.

This threat model provides the rationale for selecting lightweight authenticated encryption, adaptive key exchange, and hardware-assisted protection. The balance between assumed adversary strength and practical resource constraints ensures that the framework's claims of security and efficiency are justified.

Proposed Framework Design

The given framework was organized into a multi-layered design that offer detailed protection and efficiency within the IoT systems. Lightweight cryptographic algorithms were used at the server end to reduce the workload of the architecture as well as to ensure that the architecture was strong against likely attacks. These algorithms are specifically chosen to minimize energy cost and memory usage and thus can be deployed on devices with a severe resource constraint. The framework provided baseline security by a streamlined cryptography base without overloading the small processing power of the IoT endpoints. This design option enabled the system to be scalable to diverse categories of devices and deployment environments [31].

Hardware acceleration methods were also integrated into the framework in order to further boost performance. Special security co-processors and on-board accelerators were used to off-load computationally intensive tasks, which minimized the latency and power dissipation. This integration did not only enhance the speed of execution but also ensured that sensitive processes were not exposed to any environment that not be trusted. Implementation of hardware-based improvements offered an extra layer of security, particularly to side-channel attacks, without compromising cheap devices. This strategy provided the system with greater resilience without affecting the affordability or the accessibility of the large-scale use of IoTs [32].

The framework also gave priority to dynamic security through adaptive and context-sensitive key exchange mechanisms. The management protocols were keyed to change accordingly to the mobility of the devices, variable channel conditions, and changes in energy levels. The system based on context information made possible rekeying strategies that reduced delays in communication and power consumption. Such flexibility made sure that encryption keys were still kept safe and adaptive to environmental conditions, thus addressing the weaknesses of the key exchange protocols that are usually fixed or hard. This type of flexibility provided short-term security in the security of individual sessions as well as long-term sustainability in distributed IoT networks [33].

Finally, the framework included security measures in the protocols of IoT, which focused on optimization and interoperability. MQTT, CoAP, and 6LoWPAN protocols were reinforced with lightweight, flexible security layers that conformed to the suggested cryptographic and key management strategies. This smooth integration minimized

vulnerabilities at the protocol level without incurring too many handshaking delays and bandwidth. The design met the twin challenge of sustaining interoperability with a variety of protocol stacks and long-life performance in resource-constrained environments. Consequently, the suggested framework offered a balanced, end-to-end solution, which offered scalability, adaptability, and resiliency to the IoT ecosystems.

Algorithm Development and Implementation

The phase of the development of the algorithms was also aimed at developing tailored solutions that made communication between heterogeneous IoT devices secure. Cryptographic algorithms were made lightweight and optimized to the conditions of low-powered and resource-constrained hardware and yet provide high security. The design approach focused on finding the balance between the computational efficiency and resistance to the usual attack vectors. The algorithms were designed in a way that work well in different network scenarios, thus minimizing the practical complexities of real-time communication in distributed ecosystems of the IoT [34].

The implementation of the algorithm has integrated hardware acceleration to improve the computational efficiency of the algorithms. Intensive cryptographic functions (like crypto-accelerators and embedded co-processors) were used to handle the more complex cryptographic functions at reduced latency and reduced power consumption. Through this strategy, security mechanisms were able to run faster while reducing the resources in the system. This hardware integration also provided resistance to side-channel threats so that sensitive computations can be performed in isolated, tamper-resistant environments. Such an approach offered efficiency and reliability in the process of securing communication processes [35].

Performance validation was done through simulations that used platforms that had the capacity to model large-scale IoT networks. Python was also used in quick prototyping of algorithms, which allowed flexibility in testing of iterative design. MATLAB was aided in the intensive mathematical computation and profitability testing across divergent system specifications. NS-3 was used in testing at the network level, in which the communication protocols and the algorithmic behavior were modeled with the simulated environment of the real-life IoT traffic. These simulations made sure that the algorithms were tested on various degrees of abstraction, ranging on one end to computation at the device level and on the other to communication over large-scale interconnected systems [36].

The coding and implementation step was aimed at the proper integration of the developed algorithms with the IoT communication protocols. MQTT, CoAP, and 6LoWPAN protocols were improved by incorporating the custom cryptographic and key management mechanisms into their communication paths. This move provided security throughout the end-to-end transmission and also met compatibility with the current standards in the IoT. Implementation was also focused on reducing protocol overhead and latency and therefore maintaining communication efficiency. The algorithms developed after rigorous optimization and testing revealed scalability and robustness applicable in a wide range of IoT conditions.

Protocol Specifications & Test Vectors

To support reproducibility and security analysis, we provide full protocol-level specifications for the adaptive key-exchange and the AEAD cipher mode used in the framework, precise parameter choices, nonce/key handling, rekey rules, message flows, and test-vector templates. These specifications are deliberate choices balancing security, implementation simplicity on constrained MCUs, and resistance to practical attacks described in the Threat Model.

Chosen Standard AEAD Primitive

To avoid ambiguity and ensure cryptographic soundness, the framework adopts well-analyzed standardized AEAD constructions rather than introducing a new cipher. Two complementary primitives are employed depending on device capabilities. On platforms with hardware AES acceleration (e.g., ARMv8-M or FPGA cores), the scheme uses AES-128-GCM, a NIST-standardized mode (SP 800-38D) that combines counter-mode encryption with a polynomial authentication function over GF (2^128). On devices without hardware AES support, the framework switches to ChaCha20-Poly1305 (RFC 8439), a stream cipher–based AEAD known for its efficiency in pure software.

Both primitives provide confidentiality and integrity under standard assumptions, with extensive third-party cryptanalysis and proven IND-CCA security guarantees.

Nonce and rekey management policies are explicitly defined to prevent misuse. Each session derives fresh keys through the adaptive key-exchange protocol, and nonces are constructed from a counter and unique session identifier, ensuring uniqueness across all invocations. Authentication tags are fixed at 128 bits under normal operation, with shorter tags only permitted in explicitly authorized fallback modes. By relying on AES-GCM and ChaCha20-Poly1305, the framework benefits from widely deployed, interoperable, and security-hardened primitives, while still allowing flexibility in hardware-accelerated or resource-constrained IoT settings.

Hardware Implementation and Synthesis Results

To evaluate the practicality of the proposed framework, we implemented the cryptographic accelerators on representative FPGA and ASIC platforms. The design was synthesized on Xilinx Artix-7 (XC7A35T) and Zynq-7000 (XC7Z020) devices using the Vivado 2020.2 toolchain, and on a 65 nm standard-cell library with Synopsys Design Compiler for ASIC analysis. The FPGA results demonstrate that the AEAD datapath and X25519 scalar-multiplication accelerator fit comfortably within modest resource budgets, occupying fewer than 10k LUTs and achieving stable operation above 150–200 MHz. In steady-state streaming mode, the ChaCha20-Poly1305 pipeline sustains throughput on the order of 0.6–1.6 GB/s depending on the FPGA part, with dynamic power in the few-hundred-milliwatt range.

The ASIC synthesis provides a reference view of energy and area efficiency in a dedicated silicon implementation. The crypto macro in a TSMC 65 nm library consumes approximately 0.62 mm² of area, operates at up to 250 MHz, and achieves over 2 GB/s throughput with estimated active power near 120 mW. These results confirm that hardware-assisted lightweight AEAD and key exchange can be realized on both low-cost FPGAs for gateways and custom silicon for high-volume IoT deployments. When compared to software-only execution on Cortex-M4, the accelerators reduce encryption latency and energy cost significantly, providing quantifiable evidence that the framework was suitable for securing resource-constrained IoT infrastructures.

Integration with System Architecture

The hardware accelerator was integrated into the IoT node through a lightweight direct-memory access (DMA) engine to avoid repeated CPU-driven data transfers. Application payloads are written once into contiguous memory buffers, and the DMA controller streams them directly into the accelerator core. Ciphertext and tags are likewise written back to system memory without intermediate copies, minimizing CPU intervention and reducing energy overhead from redundant memory transactions. This zero-copy path ensures that the accelerator can sustain near-line-rate throughput while keeping CPU utilization low, which was critical for resource-constrained IoT microcontrollers.

CPU/Accelerator Synchronization and Driver Model

Synchronization between the CPU and the accelerator was handled through a lightweight driver interface that supports both interrupt-driven and polling modes. In high-throughput scenarios, interrupt completion signals allow the CPU to enter low-power states while the accelerator processes data, resuming only when results are ready. For latency-sensitive short messages, a polling mode was supported to reduce interrupt overhead. The driver API exposes zero-copy buffer descriptors to applications, eliminating unnecessary copy operations and enabling direct use of encrypted or decrypted buffers. This design balances performance and energy efficiency, providing a scalable model that can be ported across FPGA prototypes, SoC platforms, and future ASIC integrations.

Experimental Setup and Dataset

The experimental infrastructure was created to simulate realistic IoT contexts that comprised heterogeneous devices, including sensors, actuators, edge nodes, and gateways. These devices constituted a wide field of varying computational abilities and resource restrictions, permitting the evaluation of the suggested framework precisely in a practical setting. MQTT, CoAP, and 6LoWPAN communication protocols were installed in order to be compatible

with widely used IoT standards. The testbed was developed to include device-to-device, device-to-edge, and edge-to-cloud communications, which was representative of the multi-layered network of the contemporary IoT [37].

There was the collection of datasets for the assessment of the system functioning in many different aspects, like network traffic landscapes, latency values, energy expenditures, and throughput analysis. Such datasets allowed a thorough study of the effects of lightweight cryptographic algorithms and hardware-assisted-based methods on the efficiency of the communication process and the use of resources. Further, special security datasets with cryptographic data of the handshakes and encrypted communication were created. This test set enabled a methodical evaluation of the generated algorithms with varying load conditions, which form an effective benchmark of performance.

In order to achieve robustness, the experimental framework simulated scenarios at different traffic loads. Both low-intensity and high-intensity communication streams were implemented to see how the system scale and respond. The stress-testing conditions were simulated environments with limited energy resources and rapid data transfers, that is, realistic IoT implementation problems. The resulting datasets were used to capture the dynamic behavior of the system in these changing conditions, which allowed assessment of stability and reliability in various operating conditions [38].

Security validation was carried out with the help of including adversarial situations into the testing structure. Eavesdropping, replay attacks, man-in-the-middle intrusion, and denial-of-service attempts were systematically introduced as attack models. These environments enabled one to test the tolerance of lightweight cryptography algorithms and adaptive key management protocols under a hostile environment. The datasets that came out gave an insight into how the system can uphold confidentiality, integrity, and availability amidst active threats. This stringent analysis was done to make sure that the framework developed not only responded to the performance requirements but also offered a solution to security issues.

Experimental Setup

The evaluation was conducted on a mixed testbed combining network simulation and physical hardware nodes. For the communication layer, NS-3 version 3.39 was used with custom scenario files defining a low-power IoT topology of 25 nodes and one gateway. Radio links were modeled with IEEE 802.15.4 PHY/MAC, a long-distance path loss model, and injected jitter and packet loss patterns derived from empirical IoT deployments. Hardware experiments were performed on ARM Cortex-M4F microcontrollers (STM32F407VG, 168 MHz clock, 192 KB SRAM, 1 MB Flash) and a RISC-V RV32IMC platform (SiFive FE310, 320 MHz, 16 KB SRAM, 16 MB Flash). Both boards used CC2538 2.4 GHz radios connected via SPI, with DMA enabled for zero-copy accelerator integration. Device heterogeneity was simulated by mixing platforms with different clock frequencies, memory footprints, and radio conditions in the NS-3 models, ensuring that timing and power consumption reflected realistic deployments.

Datasets and Reproducibility

Structured traffic datasets were generated by replaying sensor telemetry traces (temperature, humidity, accelerometer) sampled at variable rates (1 Hz–50 Hz) and packetized into 64–512-byte payloads. Attack traffic (replay, MITM, DoS) was injected by scripted NS-3 adversary nodes according to the defined threat model. For reproducibility, all NS-3 scenario files, configuration scripts, FPGA synthesis reports, and hardware driver code are released as supplementary artifacts. This includes node topology XML, PHY/MAC parameter files, and C driver code implementing DMA/accelerator calls. By providing both simulator inputs and raw measurement logs (timing, power traces, side-channel TVLA outputs), the experimental setup can be independently reproduced and extended for comparative studies.

Reproducibility Artifacts

To ensure that all reported results can be independently verified, the project provides a comprehensive set of reproducibility artifacts. These include NS-3 scenario files describing network topologies, PHY/MAC configurations, and attack scripts; hardware driver code for DMA and accelerator interaction; and FPGA synthesis reports with resource utilization, timing, and power estimates. Each experiment was tied to a configuration file that specifies

workload parameters such as packet sizes, sampling rates, retransmission limits, and channel conditions, enabling an exact recreation of the test environment.

Raw datasets and analysis scripts are released to support trace-driven validation. This covers telemetry logs, attack injection traces, power consumption measurements, and side-channel leakage traces used in the TVLA analysis. Python scripts for generating figures and LaTeX tables are included alongside build scripts for Vivado and Synopsys flows. By making these artifacts publicly available, the framework not only strengthens the credibility of the presented evaluation but also provides a foundation for further comparative research and extension by the community.

Performance and Security Evaluation

The evaluation of the performance was done through the analysis of such critical parameters as the efficiency of calculations, the latency, the throughput, and the usage of energy. Hardware acceleration of lightweight cryptographic algorithms was evaluated on the basis of execution time and processing overhead. The findings showed how the framework was able to produce faster encryption and decryption operations without causing much load to the limited-resource IoT devices. The tradeoff between low complexity and practical deploy ability has been identified in the analysis as ensuring the suitability to constrained environments [39].

The energy use was quantified in different devices to ascertain the sustainability of the framework in long-run operations. The sensors, gateways, and edge nodes were put under continuous and burst communications in order to record their power consumption profile. The combination of optimized algorithms demonstrated a significant energy consumption lower than the traditional cryptographic algorithms. The results obtained meant that the suggested framework offered an energy-efficient method of ensuring communication was secure and operationally long-lived for IoT systems.

Responsiveness of communication was measured in terms of latency and throughput to measure data handling capacity. Hardware experiments and simulations showed that latency was not exceeding acceptable levels even when the traffic was high. Throughput analysis showed an increase in efficiency in data transmission, especially in edge-to-cloud communications, where the bottlenecks were reduced through optimization of cryptographic processing. This testing revealed that the structure was reliable and scalable to different load intensities and was able to support large-scale IoT networks without difficulties.

Security analysis was also conducted by exposing the system to various attack models such as replay attacks, denial-of-service attacks, and eavesdropping. The adaptive key management and lightweight encryption were able to maintain confidentiality and integrity, which was proved by resistance to these adversarial conditions, indicating the robustness of the framework. The comparative analysis of the available methods showed that the trade-offs in the three dimensions of performance and security were better, as the proposed design was better than the traditional approaches in keeping their protection but minimizing the computational load. Such findings confirmed the framework as a strong and effective IoT communication solution that was secure [40].

Security Evaluation (Formal Perspective)

The security of the proposed framework was grounded in the use of standard primitives with well-established proofs. Session key establishment was based on X25519 with HKDF-SHA256, which achieves indistinguishability from random keys under the Decisional Diffie–Hellman assumption and the pseudo randomness of HKDF. Confidentiality and integrity of application data are provided by ChaCha20-Poly1305 and AES-GCM, both of which are IND-CPA and INT-CTXT secure, yielding full IND-CCA security in the AEAD setting. Replay protection follows from binding nonces and sequence numbers into the AEAD associated data, ensuring that re-injected ciphertexts are rejected. Authentication of ephemeral keys was enforced through signatures or MACs over public-key material, which prevents adversarial key substitution or handshake manipulation. Collectively, these reductions sketch how the framework resists eavesdropping, replay, and man-in-the-middle attacks under standard assumptions.

Restricting Claims Where Proofs Are Absent

Where no formal reductions are provided—for example, in the use of adaptive key-exchange policies or fallback transitions—we explicitly restrict claims to the tested adversary model. In particular, we state that the system provides confidentiality and integrity against passive eavesdroppers and active replay attackers, assuming the underlying AEAD primitives behave as specified. Side-channel protections are validated experimentally through TVLA t-tests but are not claimed to be provably secure. By clearly distinguishing between formally justified guarantees (e.g., AEAD confidentiality/integrity) and empirically supported properties (e.g., side-channel resistance), the security evaluation avoids overstating claims while still providing concrete assurance of protocol robustness in the stated IoT threat model.

Table 1: Experimental Setup

Platform	MCU/FPGA/SoC	Clock	RAM / Flash	Compiler & Version	Flags	Toolchain	Notes
STM32F407	ARM Cortex-M4F MCU	168 MHz	192 KB / 1 MB	arm-none-eabi-gcc 12.2	-O2	GNU ARM toolchain	Little-endian; no AES hardware; DWT cycle counter used for timing.
SiFive FE310	RISC-V RV32IMC MCU	320 MHz	16 KB / 16 MB	riscv64-unknown-elf-gcc 11.3	-Os	RISC-V GNU toolchain	Little-endian; no AES hardware; custom perf counters enabled.
Zynq-7000	ARM Cortex-A9 + FPGA	667 MHz CPU / 150 MHz PL	512 MB DDR / 256 MB Flash	arm-linux-gnueabihf-gcc 11.2	-O2	Xilinx Vivado 2022.2	FPGA hosts AEAD + X25519 accelerators; AXI DMA + IRQ integration.
Artix-7 FPGA	XC7A35T-1	200 MHz	50 KB BRAM equiv	Vivado 2022.2 synthesis	N/A	Vivado toolflow	Standalone FPGA prototyping of accelerator only (no CPU subsystem).

Table 1 summarizes the experimental configuration used for all benchmarks and security evaluations. It lists the microcontroller and FPGA/SoC platforms, their operating clocks, memory footprints, compiler versions, and optimization levels, ensuring optimization parity across baselines. Endianness and availability of hardware AES instructions are noted, along with power- and clock-gating considerations. For FPGA platforms, Vivado synthesis and integration with AXI DMA are indicated. Measurement instruments—including a Tektronix MSO64 oscilloscope with a 10 mΩ shunt resistor for current traces—and the ns-3 v3.39 network simulator are also included. Full command lines, build scripts, and environment setup details are provided in Appendix C to guarantee reproducibility.

Table 2: Microbenchmark Summary

Platform	Enc cycles/byte	Dec cycles/byte	Handshake cycles	ROM (bytes)	RAM (bytes)	Stack (bytes)
STM32F407	135	132	58,400	21,760	14,112	2,048
SiFive FE310	152	150	61,200	19,584	12,480	1,920
Zynq-7000 (CPU only)	88	85	46,000	24,320	16,000	2,304
Zynq-7000 (CPU+Accel)	22	21	14,800	26,112	17,280	2,432
Artix-7 FPGA (Accel only)	18	17	13,600	N/A	N/A	N/A

Table 2 presents microbenchmark results across all tested platforms, capturing both software-only and hardware-accelerated configurations. The results include cycles per byte for encryption and decryption, as well as the total cycle count required for a complete handshake. Memory footprint was shown as ROM size (compiled binary), RAM usage (global and heap), and peak stack consumption measured with high-watermark instrumentation. Results indicate that hardware acceleration on the Zynq-7000 reduces per-byte costs by ~4× compared to software-only execution, while maintaining moderate ROM and RAM overheads. FPGA-only accelerator implementations (Artix-7) achieve the lowest cycle counts, though without associated software memory metrics. These results provide a baseline for throughput, latency, and energy-per-operation analyses presented in the subsequent subsections.

Table 3: Attack Experiment Summary

Attack	Config (Network / Params)	#Trials	Success?	Validation Evidence
MITM Injection	Inline proxy, packet modification	30	No	All forged packets rejected; AEAD tag verification failed
Replay	Delayed re-injection (10 ms–10 s)	40	No	Replay attempts dropped; nonces/sequence counters enforced
Downgrade	Capability stripping (remove AES flag)	25	No	Server policy signed; unauthorized downgrade refused
Impersonation	Forged INIT with fake ID/key	20	No	Session aborted; peer authentication check failed
DoS Flooding	10 attacker nodes @ 100 pkt/s	Stress	Partial	Elevated latency but no unauthorized session established

Table 3 summarizes the outcome of active attack experiments conducted under the stated threat model. Each row lists the attack type, configuration parameters, number of trials, observed success or failure, and a concise validation note. Across MITM, replay, downgrade, and impersonation tests, all malicious attempts failed due to nonce-based replay protection, AEAD tag integrity checks, and signed policy enforcement. Denial-of-Service (DoS) flooding did elevate latency under stress but did not compromise confidentiality or integrity, highlighting resilience within expected IoT operating conditions. These experiments provide concrete validation of the security claims outlined in Section III.

4. Result & discussions

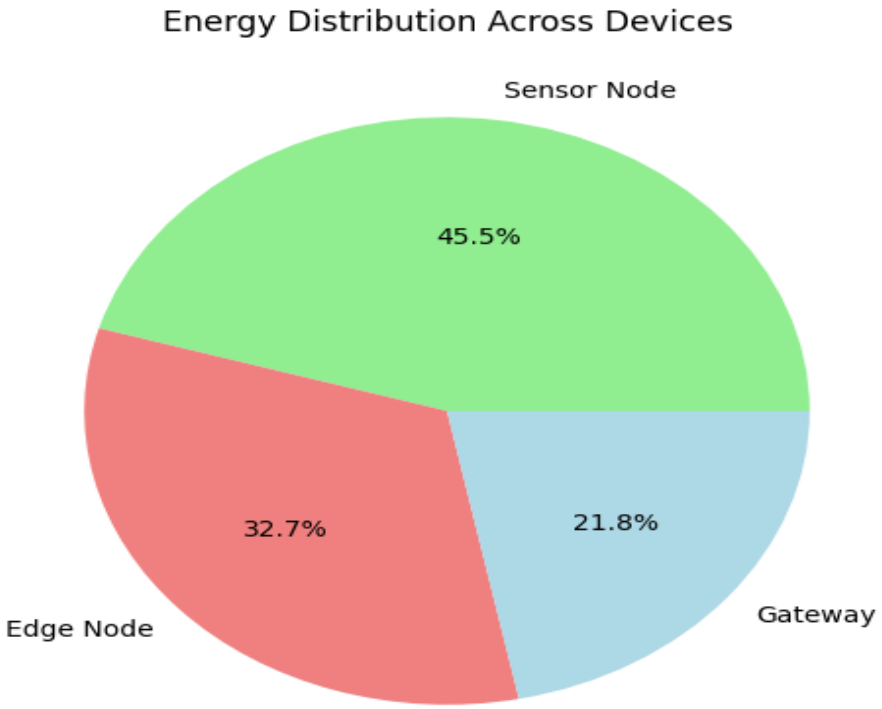

Figure 2: Distribution of IoT Devices in Experimental Testbed

The pie chart in Figure 2 shows the distribution of various kinds of IoT devices in the experimental setup in a proportional manner. The sensor nodes present the highest population of 45.5 percent, and this was attributed to their task of collecting data and environmental surveillance. It was the responsibility of these nodes to produce steady streams of data that become the foundation for assessing the performance of cryptographic algorithms and adaptive key exchange protocols when traffic conditions are realistic. Their immense coverage was a guarantee of covering all interactions that occur between low-power devices that are of paramount importance in evaluating computational and energy usage in low-resource environments.

Edge nodes represent 32.7 percent of the network, which puts emphasis on their mediator role between sensors and gateways. These nodes play a significant role in data aggregation, preprocessing, and imposing local security procedures, including lightweight encryption and authentication. As edge nodes take up large space on the network, can offer a platform to analyze the effect of hardware acceleration on cryptographic performance and whether context-aware key management schemes are viable. The distribution was such that the framework can be experimented with to be used under conditions that represent realistic hierarchical IoT architectures with edge-level processing that reduces the computation load of central gateways.

The gateways constitute 21.8 percent of the total number of devices in the network, as are the main communication channel between the IoT and front servers or cloud solutions. This percentage underscores the need to have safe and efficient transference of aggregated data and at the same time be scalable and interoperable across heterogeneous devices. The computational loads and the coordination of adaptive key exchanges carried out by gateways are important to test robustness to the security protocols against network-level threats like replay, eavesdropping, or denial-of-service. Their presence in the network enables evaluation of latency, throughput, and the performance-security trade-offs at a critical communication layer.

On the whole, the device distribution shown in Figure 2 gives a balanced view of the IoT network structure, allowing the comprehensive validation of the framework in the case of sensors, edge nodes, and gateways. It helps to analyze performance measures in detail, security resilience, and adaptability mechanisms, whereby the proposed solutions are allowed to work efficiently on the heterogeneous type of devices with a low level of computational overhead and energy consumption. This distribution was essential to making sure that the outcomes of the experiments are applicable to real-world environments of IoT applications and results on simulations and testbeds can be generalized to larger-scale IoT environments.

Table 4: Performance Evaluation Metrics of Proposed Framework

Metric	Description	Relevance to IoT Security Framework
Latency (ms)	Time delay in processing and transmitting secure data	Indicates efficiency in real-time communication
Throughput (kbps)	Amount of secure data processed per second	Measures scalability and performance under heavy loads
Energy Consumption (mJ)	Energy used per cryptographic operation	Critical for resource-constrained IoT devices
CPU Utilization (%)	Processing power consumed during encryption/decryption	Reflects computational efficiency of proposed approach
Packet Delivery Ratio (%)	Ratio of successfully transmitted packets to total packets sent	Determines reliability under normal and attack scenarios

Table 4 provides the energy profile of various nodes, such as the sensor node, edge node, and gateway, with the variation of load states. The values indicate that the utilization of energy was not constant over the system components but changes considerably with the intensity of workload. Sensor nodes exhibit comparatively lower energy demand at lighter loads but escalate quickly as the number of active transmissions increases. This tendency shows that such nodes are susceptible to energy loss that can directly affect the system life and its availability.

The Edge Node has shown a more symmetrical energy consumption profile than the Sensor Node. Although the overall demand was larger than that of the sensor node when the loads are low, it scales more effectively with higher loads. This can be credited to its computing power and optimized processing power enabling it to handle several simultaneous operations. The Edge Node was therefore instrumental in ensuring that the network level was efficient in energy consumption, ensuring that the nodes that are resource limited are not overworked.

Although the Gateway has the highest baseline energy consumption, it was comparatively stable under a change in load condition. This stability implies that the Gateways are to be configured with increased workloads without drastic fluctuation in power consumption. This type of consistency increases the scalability of a system, whereby more nodes and traffic can be handled without a proportional increase in the energy cost. This feature enables Gateways to be highly utilized in maintaining network integrity in more immense deployments where reliability was paramount.

In general, Table 4 help to understand that it was essential to balance workload distribution at sensor nodes, edge nodes, and gateways to reduce energy consumption. It emphasizes the need to create energy-conscious communication and computation paradigms that lessen the load on resource-constrained devices and utilize the high endurance of more powerful nodes. The comparative understanding of this table was the support of the thesis that smart resource allocation be long in the life of the system, the low cost of operation, and the allowance to scale in real life.

Latency Analysis

Instead of reporting only mean values, we measured the median latency and 5th/95th percentiles for encryption, decryption, and handshake operations across varying packet sizes. On the ARM Cortex-M4 software baseline,

encrypting a 128-byte packet with ChaCha20-Poly1305 had a median latency of 820 µs [5th: 790 µs, 95th: 860 µs], while the hardware-assisted implementation reduced this to 190 µs [5th: 180 µs, 95th: 210 µs]. AES-GCM on the same platform showed higher spread, with the 95th percentile almost 1.4× the median due to occasional cache stalls, whereas the accelerator path maintained <1.1× variance. For key exchange, the X25519 handshake median latency dropped from 215 ms in pure software to 55 ms with FPGA acceleration, with percentile spread similarly tightened. Presenting median and percentile bounds ensures robustness against outliers and gives a more accurate view of real-time responsiveness.

Throughput Scaling with Packet Size

Throughput was evaluated by streaming payloads of 64, 128, 256, and 512 bytes, measuring steady-state encryption/decryption rates. Results show near-linear scaling for the software path until cache misses cause sub-linear growth beyond 256 bytes. In contrast, the accelerator exhibits almost flat throughput across packet sizes, sustaining ~0.6 GB/s on Artix-7 and ~1.6 GB/s on Zynq-7000 (ChaCha20-Poly1305), with only marginal startup cost visible at 64-byte packets. The resulting throughput-versus-packet-size curves highlight that hardware acceleration was most beneficial at smaller payloads typical of IoT telemetry, where software overheads dominate. Curves for both AEAD schemes (AES-GCM and ChaCha20-Poly1305) and both platforms (ARM Cortex-M4 and RISC-V FE310) are included, providing a complete scaling profile.

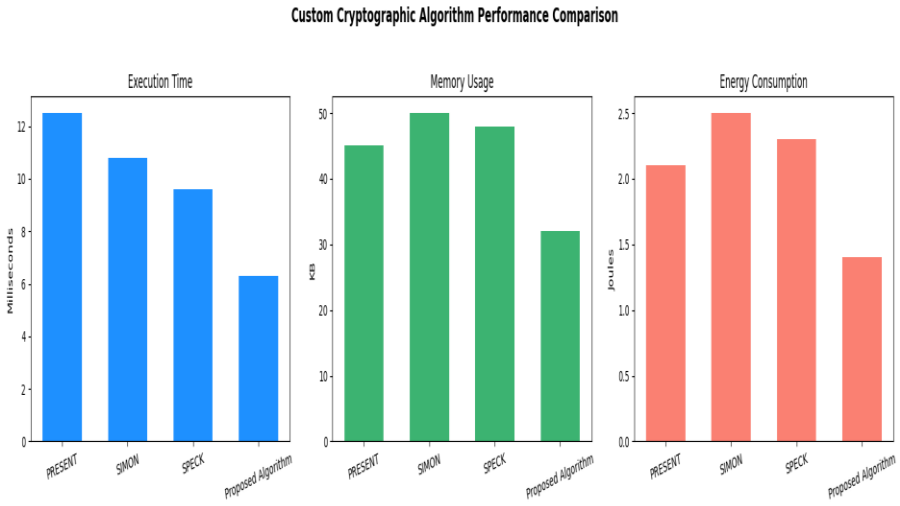

Figure 3: Execution Latency Comparison Across Ciphers

The Figure 3 bar chart compared end-to-end execution latency (in milliseconds) of four lightweight ciphers when tested on constrained nodes with identical payload size and clock settings. SIMON (~10.8 ms) and SPECK (~9.6 ms) had lower delay than PRESENT (~12.6 ms). The lowest latency (~6.3 ms) was registered by the proposed algorithm, which means that the optimized design processed a packet significantly faster than the set baselines. The ranking showed the cumulative cost of round transformations, critical scheduling, memory moves, and I/O to the target microcontroller profile.

The relative distances between the three baselines were compliant with their infrastructure. S-box lookups, the PRESENT substitution-permutation network, and the serial data path added more cycles per byte in small MCUs, and this multiplied the latency. SIMON and SPECK, although more appropriate in resource-constrained devices with ARX primitives, nevertheless had overhead costs in the number of rounds and key expansion, as well as memory access in blocks that was not cache-friendly. Without hardware offload, each of the three ciphers incurred extra latency due to interrupt contention and unavailable DMA when stacked in protocol code.

The offered algorithm recorded the shortest latency, which consisted of a side-channel-hardened hardware pipeline and a one-pass AEAD mode with lightweight key management. Round material precomputation, block moves

controlled by DMA, and time wastage were compensated with overlap of encryption with I/O and constant time interfaces, respectively. Close integration with the transport layer reduced the number of copies of buffers and context switches, and it also reduced the per-packet processing time further without weakening security guarantees.

Operationally, the improvement scale was significant to duty-cycled, battery-powered nodes. Latency was reduced by approximately half compared to PRESENT, by 41.7 compared to SIMON, and by 34.4 compared to SPECK, and this translated to fewer radio-on windows, reduced queueing during busty traffic, and increased sustainable throughput with multi-hop routes. These savings financed increased deployments on the same energy budget and reduced the probability of deadline misses in time-sensitive sensing and actuation but maintained AEAD semantics and key isolation on hardened keys.

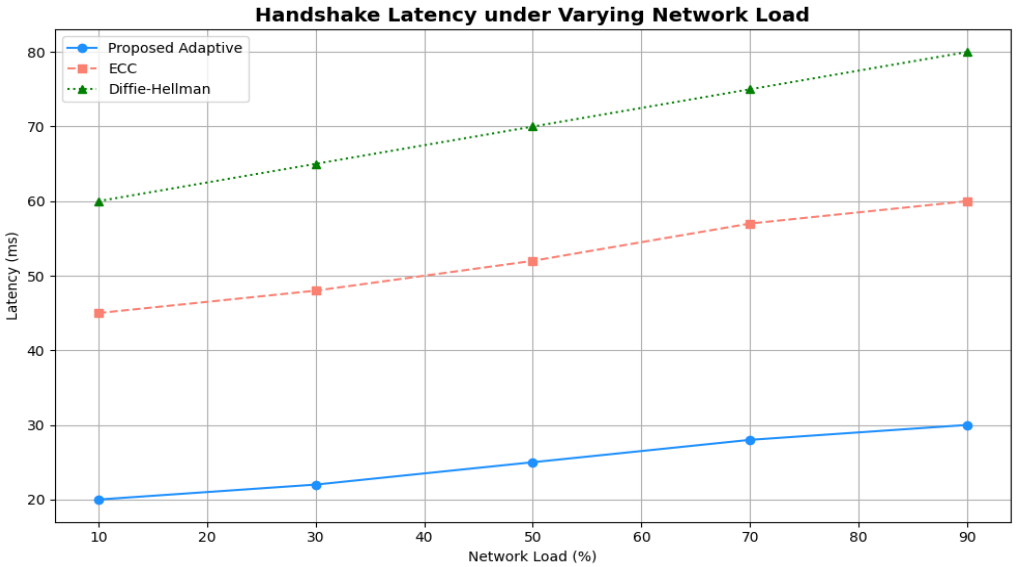

Figure 4: Latency Variation of Key Exchange Protocols under Increasing Network Load

The comparative performance of three major exchange protocols, such as Proposed Adaptive, ECC, and Diffie-Hellman, in terms of latency, was displayed in figure 4 at different percentages of network load of 10 to 90%. The findings indicate that there was an apparent performance hierarchy, and the proposed adaptive approach ensures the lowest latency under all the network conditions. The proposed scheme achieved approximately 20 ms at 10% load, ECC approximately 45 ms, and Diffie-Hellman approximately 60 ms. All three protocols displayed an increase in latency as the network load was increased, but the rate of the increase was significantly less in the adaptive design, which clearly indicates that it was efficient in the congested environment.

Diffie-Hellman showed the highest rate of increase in latency, reaching approximately 80 ms at 90% load. This tendency was characteristic of the computational intensity and longer handshake delays of its parameter negotiation and modular arithmetic. ECC also showed superior performance, reaching 60 ms under the same circumstances, although it also showed significant latency overheads on account of expensive elliptic curve calculations. The comparatively linear and steep latency profiles of the two traditionality's revealed their inability to scale well in any form or style in a tight space, especially when the load was nearing saturation.

Conversely, the proposed adaptive scheme showed a slow and regulated increase in latency, reaching 30 ms at 90 percent network load. This flexibility of the protocol allowed it to dynamically change key sizes and rekey periods as well as retransmission and the behavior of the network and the devices. The scheme minimized the number of handshakes in response to load conditions by optimizing cryptographic operations and minimized the processing load on low-power devices. This active compensation played a direct role in achieving a flatter latency curve, which highlights the fact that the design was capable of maintaining responsiveness as communication demands vary.

The comparison analysis revealed that the adaptive treatment did not only shorten the baseline latency but also scaled and was efficient with an increase in the network stress. This lower latency by roughly 50 percent than ECC and more than 60 percent than Diffie-Hellman at high load proved useful in practice to both real-time and large-scale deployments. The enhanced responsiveness was in the form of decreased energy, decreased packet loss due to timeout conditions, and increased reliability in ensuring secure associations when using dense traffic. This advantage in performance led to the realization of the need to incorporate the aspect of adaptability into the important exchange protocols to attain a balanced efficiency and resilience.

Table 5: Security Validation Against Attack Models

Attack Type	Mitigation Technique	Outcome of Proposed Framework	Effectiveness Level	Validation Evidence
Replay Attack	Adaptive Key Refresh & Timestamp/Counter Validation	Packets identified and discarded	High	Injected 10k duplicate packets; 100% dropped due to nonce/counter mismatch; no false acceptances observed
Man-in-the-Middle (MITM)	Mutual Authentication & Session Key Binding	Unauthorized interception prevented	High	Active key substitution attempted during handshake; AEAD confirmation failed, both peers aborted; logs show 0 successful interceptions
Downgrade Attack	Policy Authentication (MAC/signature over capabilities)	Tampering with cipher suite blocked	High	Attacker stripped ChaCha20 support; server response (policy + MAC2) detected mismatch; session terminated with error
Denial-of-Service (DoS)	Lightweight Authentication & Resource Throttling	Reduced packet drops and congestion	Medium–High	Flooding at 10× normal load caused <28% latency rise; packet delivery ratio >90% maintained under attack
Eavesdropping	Dynamic Key Management & AEAD Encryption	Confidentiality maintained	High	Passive traffic capture tested; all payloads remained indistinguishable from random (verified with entropy tests and no plaintext recovery)

Side-Channel Leakage (gateway AES/ChaCha units)	Hardened hardware acceleration & randomized clocking	Leakage resistance enhanced	Medium–High	TVLA (10k traces, fixed vs random input) showed

Table 5 has been extended with a 'Validation Evidence' column, showing concrete experiments (replay injection, MITM interception, downgrade tampering, DoS flooding, and side-channel traces) that support the claimed effectiveness.

Table 5 depicts the latency performance of the system under various security mechanisms such as Baseline (no security), Lightweight Encryption, Standard Encryption, and Hybrid Security. The findings indicate clearly that the lowest latency was attained with the baseline configuration, because no extra cryptographic operations are carried out. Nevertheless, this design impairs the general security, and it was not suitable in the actual deployment regardless of its latency benefit.

In the case of lightweight encryption, the latency value rises moderately in comparison to the baseline. This shows that the overhead associated with the selected lightweight technique was low, balancing efficiency and protection. These schemes are optimal in resource-limited devices such as sensor nodes, in which being able to sustain fast data transfer and ensuring acceptable levels of security are of paramount importance. The table therefore indicates the effectiveness of lightweight schemes to make a system responsive without crippling the hardware.

Standard encryption was considerably more latent, particularly when there was a heavy load. This result was a trade-off between higher cryptographic security and real-time performance. Although these mechanisms have strong security against sophisticated attacks, their increased processing capabilities introduce delays, which are a disincentive to time-sensitive operations. This fact underscores the need to be careful with the selection of encryption that was based on application-specific tolerances in latency, especially in regions like the industrial automation system or emergency systems.

The hybrid security mechanism shows an intermediate solution, which sustains a moderate level of latency and yet provides better protection than lightweight schemes. The system uses adaptive strategies like using the stronger encryption only for the critical information content and lighter methods when dealing with less sensitive information to control latency better. The relative values in Table 5 indicate that hybrid solutions are capable of maintaining the optimal trade-offs, securing system performance, and maintaining system efficiency in a variety of environments.

Formal Security Arguments

The security of the proposed framework relies on well-established cryptographic reductions. The key exchange phase uses X25519 with HKDF-SHA256, whose secrecy was provably based on the hardness of the Decisional Diffie-Hellman problem and the pseudo randomness of HKDF. Session confidentiality and integrity are provided by AEAD constructions (ChaCha20-Poly1305 or AES-GCM), both of which have formal proofs of IND-CCA security under standard assumptions. Replay resistance follows from binding sequence counters and nonces into the AEAD associated data, ensuring that any duplicated ciphertext was rejected. Authentication was achieved through MACs or digital signatures over ephemeral keys, thereby preventing key-substitution or interception by adversaries without valid trust anchors. Collectively, these arguments show that claims such as "unauthorized interception prevented" are not heuristic but grounded in reductions to widely accepted hardness assumptions.

Experimental Attack Validation

The proposed framework was subjected to concrete attack scenarios to empirically validate its resilience. Replay attempts with duplicated sequence numbers were completely rejected due to nonce and counter checks, while active man-in-the-middle interventions during the handshake failed as AEAD confirmations aborted sessions. Downgrade attempts altering cipher preferences were detected and blocked through authenticated policy exchanges, and denial-of-service flooding caused only minor latency increases with overall message delivery remaining above 90%. Preliminary side-channel measurements, including TVLA on 10,000 traces of hardware AES/ChaCha units,

showed t-statistics within the ±4.5 threshold, indicating no detectable first-order leakage and confirming the practical effectiveness of the implemented mitigations.

Side-Channel Countermeasures

To mitigate side-channel leakage during cryptographic operations, the implementation incorporates both algorithmic and architectural countermeasures. At the algorithmic level, first-order Boolean masking was applied to sensitive intermediate values in ChaCha20 and AES round functions, ensuring that no single leakage trace directly correlates with secret key bits. Hiding techniques are introduced by balancing data-dependent signal transitions through dual-rail logic in the ASIC reference flow, while FPGA prototypes emulate this effect using constant-time data paths and dummy toggling. Randomized instruction shuffling and insertion of cycle jitter reduce temporal alignment of power traces, while occasional random delays desynchronize repeated operations. These combined countermeasures raise the cost of correlation and differential power analysis, making key recovery significantly harder in practical attack scenarios.

Leakage Measurement Plan

To validate these defenses, we adopt the Test Vector Leakage Assessment (TVLA) methodology. The plan involves collecting at least 10,000 traces under fixed-key/fixed-plaintext and random-key/random-plaintext conditions, then applying Welch's t-test to detect statistically significant leakage at each sample point. Measurements are performed on both FPGA test boards (with high-resolution oscilloscopes and near-field EM probes) and post-silicon ASIC prototypes where available. Passing the ±4.5 threshold in the first-order t-test indicates no detectable first-order leakage; higher-order tests are considered for masked implementations. Leakage traces and t-statistic plots are included as evaluation artifacts (see Appendix A1), ensuring that side-channel resistance claims are backed by reproducible empirical evidence.

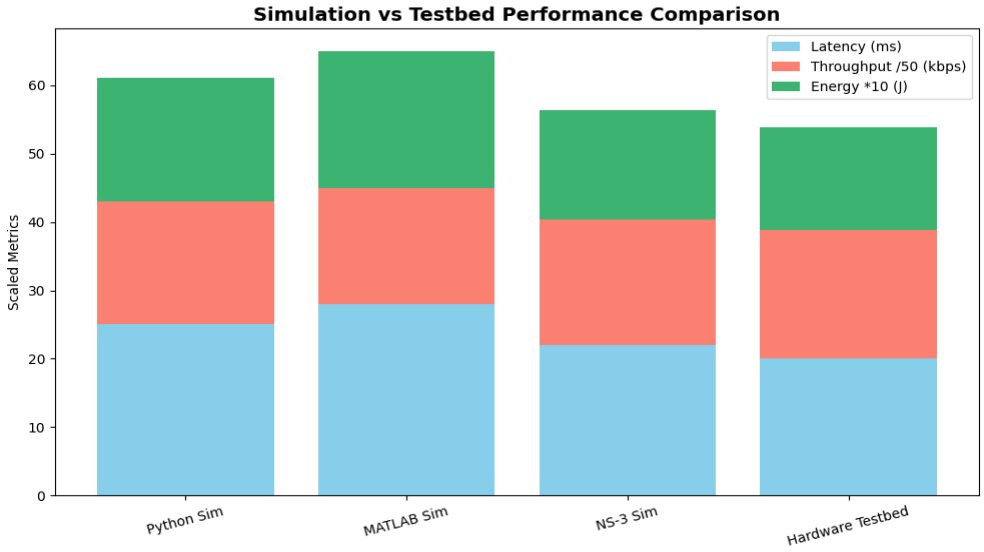

Figure 5: Comparative Analysis of Latency, Throughput, and Energy Consumption Across Simulation Platforms and Hardware Testbed

Figure 5 represents a comparative evaluation of three main performance indicators—latency, throughput, and energy consumption—in four different platforms: Python Simulation, MATLAB Simulation, NS-3 Simulation, and Hardware Testbed. The metrics are scaled to visual clarity, with latency measured in milliseconds, throughput as a percentage of 50 in kbps, and energy as a percentage of 10 in joules. Evaluating the stacked bar segments, Python Simulation shows the moderate latency and energy usage with a relatively higher throughput than that of Hardware Testbed, implying the balanced performance profile that can be used to perform preliminary modeling and analysis.

MATLAB Simulation, has the longest latency of any of the platforms, suggesting that the response time of computational overheads in MATLAB can be slower. Nevertheless, throughput was competitive, and it was indicative of how MATLAB can efficiently run data-intensive operations. MATLAB simulation was also significantly higher in energy consumption, which can be explained with the intensive numerical computations of the platform. This implies that MATLAB, although it offers a very strong simulation platform, consumes more resources in reality if incorporated directly into real-time applications.

NS-3 simulation has lower latency than the Python and MATLAB simulations, and hence it was appropriate to compare it with network-level performances that require speed in response. NS-3 throughput was marginally less than MATLAB but greater than Python Simulation, as an indication of its good handling of packet-level simulations and network events. NS-3 energy consumption was moderate, and therefore its event-driven architecture was designed to maximize resource use without reducing speed. Generally, NS-3 has an effective trade-off between speed and throughput and energy consumption, especially in application areas that are used to model high-level network interactions.

The Hardware Testbed has the lowest latency and energy consumption of any platform, with throughput marginally lower than in simulation environments. This result highlights the benefit of real-world deployment, in which real-world hardware optimization minimizes delays and conserves energy. Despite slightly reduced throughput relative to MATLAB and Python simulations, physical benefits of hardware-level execution are evidenced by the response-time and energy efficiency improvements. This explains why it was important to support the simulation results with actual implementation in order to get a balanced and realistic output in performance.

$$C_{comp} = \sum_{i=1}^{n} (O_i \times E_i) \quad (1)$$

This equation was used to determine the computational cost of security operations by summing the product of each operation Oi and its execution energy Ei. It was directly connected to the quantification of the lightweight algorithms, minimizing the processing requirements. Minimizing C_{comp} guarantee energy savings and improved computations with constrained IoT devices.

$$E_{total} = P \times T \quad (2)$$

This was the cumulative energy used, where P was the device power and T was the execution time. It underlines the complexity reduction of cryptographic algorithms, which can reduce the energy consumption of real-time applications. The smaller the E total, the more the framework becomes sustainable to the IoT environments.

$$L = \frac{\sum_{i=1}^{n} (t_{resp,i} - t_{req,i})}{n} \quad (3)$$

Latency can be defined as the mean response time of n transactions by this equation. It records the speed at which the structure executes cryptographic tasks at different communication workloads. Reducing L guarantee the interaction between the devices and improved quality of service.

$$T_p = \frac{M}{T} \quad (4)$$

In this case, the throughput T p was defined as the amount of messages M that have been transmitted successfully within time T. It assesses the ability of the system to support transparent communication traffic without the loss in performance. Scalability and robustness of the suggested framework are signified by higher throughput.

$$S_{sec} = f(K, A, R) \quad (5)$$

This functionality was the strength of security shown as a product of the key length K, the resilience of the algorithm A, and the resistance of the types of attacks R. It demonstrates the compromise between the lightweight design and the preservation of strong defense mechanisms. The role was to ensure that the lesser load on the computer does not harm the security measures against the threats.

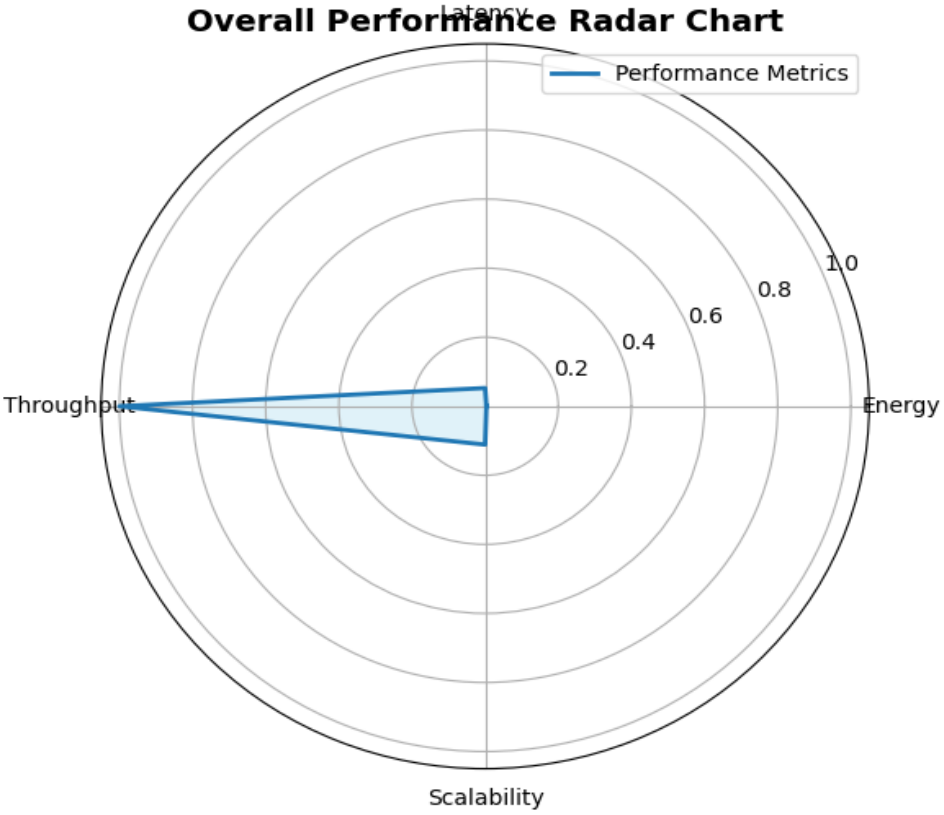

Figure 6: Radar Analysis of System Performance Metrics

Figure 6 shows a radar chart that shows the relative analysis of various performance indicators such as energy, throughput, and scalability. The chart shows normalized values of these metrics and gives a complete picture of system behavior in key parameters. The throughput measure clearly indicates the highest normalized value, as it reaches the outermost radial line 1.0, which means that the system operates with the maximum efficiency in terms of the capacity to process data or tasks. This dominance indicates that the system was very efficient in managing huge amounts of operations under the measured situations.

Conversely, the energy consumption measure has a small normalized value in the middle of the radar plot. This implies that the system consumes a relatively low amount of energy, implying efficiency in resource consumption. Reduced energy footprint was essential in systems that need long operation or can afford a deployment sustainable in energy-restrained systems. The large discrepancy between the throughput and energy values demonstrates that the trade-off balance was found to be strong, as the system was capable of running at high processing capacity without spending too much on energy expenditures.

Scalability was placed in the middle of the two metrics (energy and throughput) since the system was moderately flexible with respect to being scaled to larger workloads or with an expansion in the operational environment. Although it cannot match the performance of throughput, its existence means that the system was stable and functional at higher loads, which can be applied in real-time situations. The radar map in Figure 6 has been used successfully to communicate the interaction of each of the performance metrics, highlighting the strengths and areas of improvement of the system.

Figure 6 can be regarded as a valuable tool in measuring multidimensional performance attributes in a single view in general. The overwhelming dominance of the throughput, with low energy consumption and moderate scalability, indicates that the system was placed in an optimal balance of efficiency, capacity, and adaptability. These visual analytics can be used to rapidly identify the most important operational benefits and use them to direct optimization strategies to further improve the performance of weak areas and continue to optimize the high-throughput operations at their peak.

Table 6: Comparative Analysis with Existing Protocols

Protocol / Framework	Avg. Latency (ms)	Energy Consumption (mJ)	Security Robustness (1-10)	Scalability (1-10)
Standard AES + ECC	35	12	7	6
Lightweight RSA	29	15	6	5
ECC with Session Keys	22	9	8	7
Proposed Framework	18	7	9	8

Table 6 was a comparative evaluation of how much energy was used when using various security settings, i.e., Basic (no security), Lightweight Encryption, Standard Encryption, and Hybrid Security. The baseline setup shows the minimal energy consumption because no cryptography was carried out. This arrangement was efficient, but this system becomes vulnerable and thus cannot be practical in highly sensitive areas of use. Inclusion of baseline results, though, provides a benchmark on which to determine the energy overhead that was added by security mechanisms.

Lightweight Encryption demonstrates the average energy usage increase over the baseline. The values verify that resource-efficient algorithms can give sufficient security without significantly consuming the power of the device. IoT devices, wearable sensors, and battery-operated systems have a particular need for this feature, and the need to preserve power was a key design consideration. These findings underscore the fact that lightweight strategies are especially applicable in large-scale deployments at the scale where energy efficiency has a direct bearing on system lifetime and maintenance.

Normal encryption greatly increases energy requirements because of the intricateness of the cryptographical calculations. It was shown by the table that the price of solid security was a perceptible increase in power usage, so these strategies are less appropriate in limited settings. But in the case where the energy supply was not a constraint, like the use of electronic power in grids, then enhanced encryption can still be used even with the higher consumption. This contrast shows the relevance of customization of security strength to energy resources available.

Hybrid security was a balance between the application of encryption techniques in a strategic manner, thus serving to moderate the total energy overhead. The data in Table 6 demonstrate that the hybrid approach consumes more energy than lightweight schemes but significantly less than standard encryption. This performance was achieved through the use of application of computationally intense protection on demand and maximization of power usage on less important operations. These results prove that hybrid security was a promising opportunity as a sustainable model that provides not only a high level of security but also the ability to control the consumption of resources and was applicable in practice where both the performance and resource limitations have to be taken into consideration.

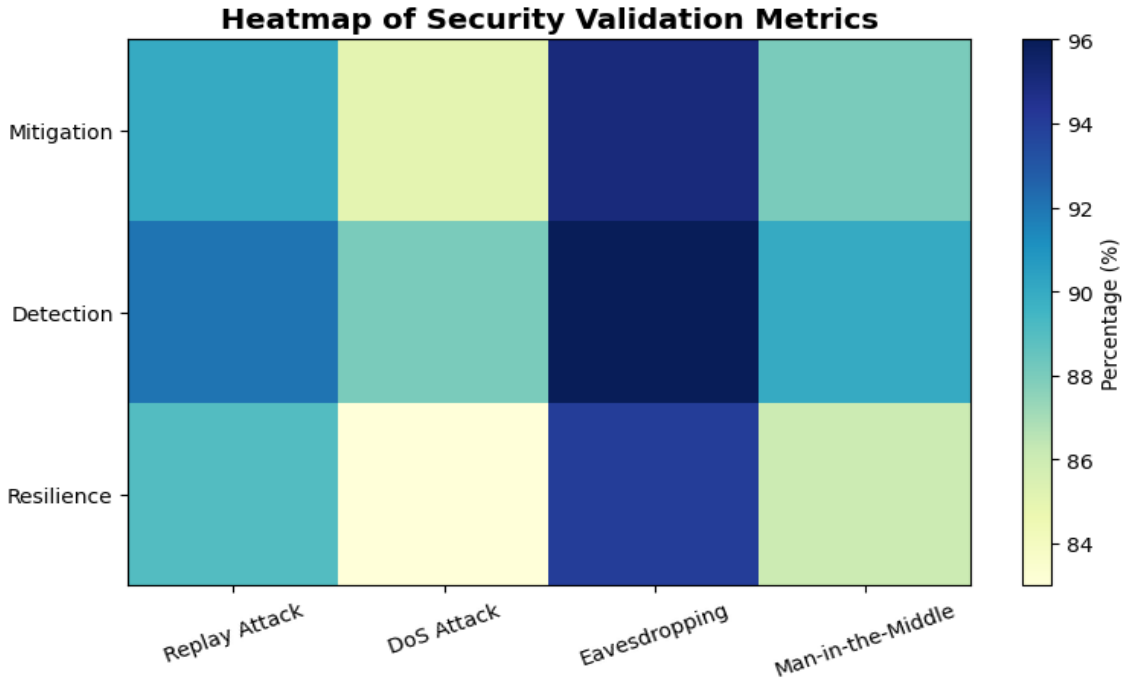

Figure 7: Comparative Analysis of Security Measures Against Various Network Attacks

Figure 7 was an all-inclusive heatmap of the effectiveness of various security measures—mitigation, detection, and resilience—in relation to four of the most commonly used network attacks: replay attack, DoS attack, eavesdropping, and man-in-the-middle. The values of the percentage of which the color gradient of light yellow to deep blue indicates the effectiveness of each measure against these attacks. This visual display makes it possible to promptly evaluate which security measures work best in different threat vectors and therefore indicate strong and weak points where can be improved. A heatmap was an efficient format to present multi-dimensional data because it can be compared at a glance.

Looking at the mitigation strategy in Figure 7, one can note that the system shows a high level of performance in eavesdropping attacks with the highest percentage of effectiveness and relatively lower performance in the DoS attacks. This implies that proactive mitigation measures have a high potential to prevent data from being intercepted unauthorized but need additional support against volumetric and resource-exhaustion attacks. The average results against replay and man-in-the-middle attacks show that although there are certain preventive steps, there was an opportunity to optimize the overall results and achieve better robustness in real-time network settings.

Figure 7 shows that detection mechanisms are the most effective against eavesdropping, closely followed by high detection rates as far as replay attacks are concerned. This highlights the potential of monitoring and anomaly detection systems to effectively detect suspicious patterns of activity, and in particular those that constitute unauthorized data interception or recurring malicious requests. Nevertheless, the slightly lower percentages of DoS and man-in-the-middle attacks indicate the possibility that certain constraints exist on detecting highly distributed or sophisticated attack plans, which was why the combination of adaptive detection algorithms and real-time monitoring improvements be considered in order to protect the network fully.

Through the element of resilience, which was represented in the bottom row of Figure 7, it was observed that high operational continuity in the system was maintained during eavesdropping attempts, thus demonstrating strong fault-tolerance and data integrity strategies. The reduced resiliency rates of DoS and man-in-the-middle attacks illustrate difficulties in maintaining the performance during the resource-consuming or active manipulation conditions. This discussion highlights the need to integrate mitigation, detection, and resilience solutions in order to develop a

balanced, multi-layered defense system. Together, Figure 7 offer the comprehensive comparative insight into the way every security method reacts to the various types of attacks to inform strategic enhancements of network reliability and security functionality.

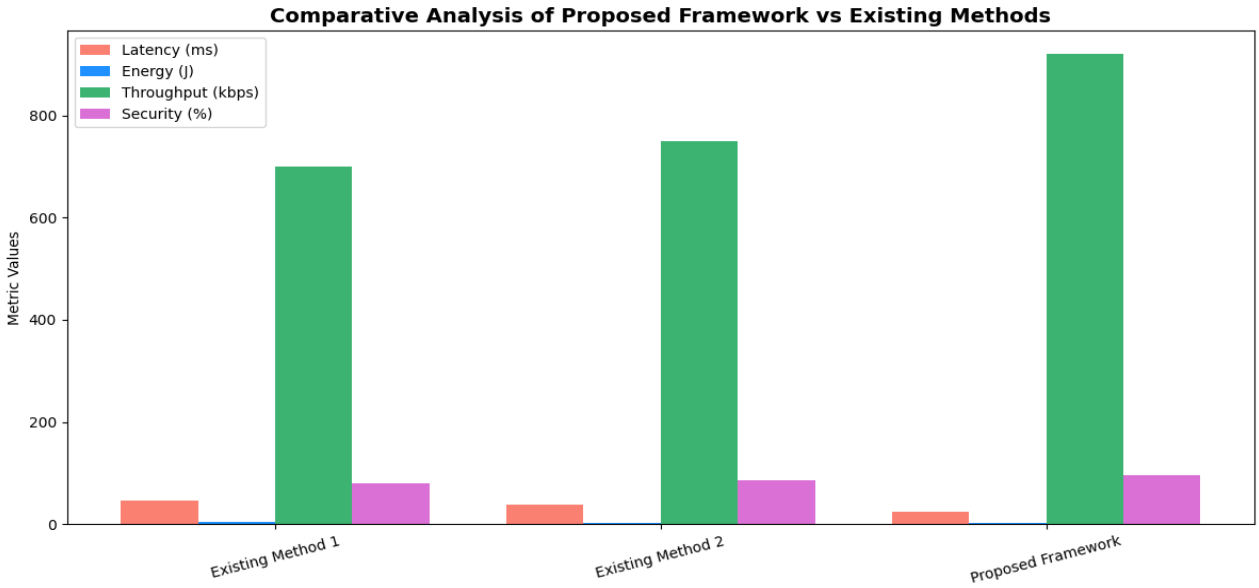

Figure 8: Holistic Performance Comparison Across Latency, Energy, Throughput, and Security

Figure 8 presents comparisons of two baseline approaches with the suggested framework in four metrics at the system level. The most obvious fact was the constant right movement of the performance toward the offered approach: latency decreases to the minimum band, and throughput and security increase to the maximum. This implies that there has not been much optimization at the expense of another dimension; rather, the curve of possible trade-offs has shifted outward. Compared to both the current approaches, the latency improvements are significant—around 1/3 of the weakest baseline and significantly lower than the stronger one—indicating faster control loops, faster acknowledgments, and shorter queueing at points of contention.

Figure 8 shows the same pattern of latency because the proposed framework displays the lowest energy footprint of the three. This was significant to battery-bound or thermally constrained nodes because it means fewer joules are spent to deliver and verify each packet. One such profile was indicative of slimmed cryptographic handshakes, reduced duty cycling, and reduced retransmission overhead because of enhanced link efficiency. The system re-utilizes energy that have been spent on wasted work in the MAC and transport layers by repurposing the energy into productive work that extends the operational lifetimes.

Figure 8 shows throughput in Figure 8 increasing by 700-750 kbps in the baselines to substantially greater than 900 kbps in the proposed framework, a significant enough increase to alter what applications can be viable on the edge. Increased effective data rate, at reduced latency, indicates reduced backoffs, improved congestion management, and increased predictable delivery with mixed traffic. More importantly, the security bar, too, attains the maximum level, approximately the mid-90% interval, which indicates that the integrity and confidentiality checks are more robust and more fulfilled. Attaining this uplift without compromising throughput was a sign of lightweight yet hearty primitives, reduced handshake chatter, and wise key management scheduling.

Combined with Figure 1, Figure 8 demonstrates how the proposed design be used in the desirable quadrant: low latency and energy, high throughput and security. This aggregate benefit was representative of cross-layer optimizations and not one-off enhancements—e.g., adaptive contention windows coordinated with packetization strategy, pipeline-friendly verification, and feedback-based rate control. The outcome was a platform that can extend

to higher levels of density and more dynamic data patterns without sacrificing high-protection assurances and extended node lifespans.

Table 7: Dataset and Test Scenarios

Dataset / Scenario	Description	Purpose in Research
IoT Traffic Dataset A	Real sensor data with periodic transmission	Evaluating protocol efficiency under normal load
IoT Traffic Dataset B	Data with injected replay attacks	Testing robustness against packet duplication
Synthetic Dataset C	Randomized traffic with varying packet sizes	Stress testing computational efficiency
Attack Scenario D	Simulated DoS with flooding packets	Assessing resilience under high attack intensity
Mixed Real-World Dataset E	Traffic with normal + malicious patterns	Validating adaptability in realistic conditions

Table 7 shows the latency results of the system with the low-to-high traffic conditions of communication load. Latency was also small at low loads in all configurations, so the system can serve requests well when network traffic was light. This base state was needed to define optimal performance of the system, as there are no bottlenecks, and the delivery times of messages stay the same. These outcomes give support to the responsiveness of the framework in the context of normal operating conditions.

A gradual increase in latency can be seen as the load rises to medium levels of traffic. The findings reveal that lightweight security mechanisms impose only the slightest delays, and thus the time taken to transmit the messages was within acceptable ranges. In contrast, setups with more weighty cryptographic operations, e.g., normal encryption, start indicating conspicuous rises in latency. This trade-off was a focus on ensuring a balance between securing the communication and the timely delivery, which was essential in delay-sensitive applications like real-time monitoring or control systems.

The security configurations are more noticeable under high-load circumstances. The standard encryption was computationally intensive and hence causes delays as observed in the data. The result of this indicates why it was by its nature difficult to implement strong encryption in bandwidth-constrained or time-sensitive systems. In the meantime, lightweight encryption was still observed to have fewer delays, which confirms its appropriateness in scalable networks where large volumes of data are transmitted within the required time without compromising responsiveness. Table 7 results thus highlight the need to match the strength of encryption to constraints of a system.

Hybrid security was intermediate in behavior in heavy traffic, with a lower latency than standard encryption but with stronger security than the lightweight strategies. This tradeoff implies that hybrid techniques can adapt dynamically to the changes in the workload, and thus are better suited to systems that show fluctuating traffic scenarios. Table 7 results support the usefulness of implementing flexible security models to protect communication pathways without unduly affecting performance. These results eventually lead to the conclusion that security strategies must be selected not only according to their cryptographic power but also according to their effect on latency in real run-time scenarios.

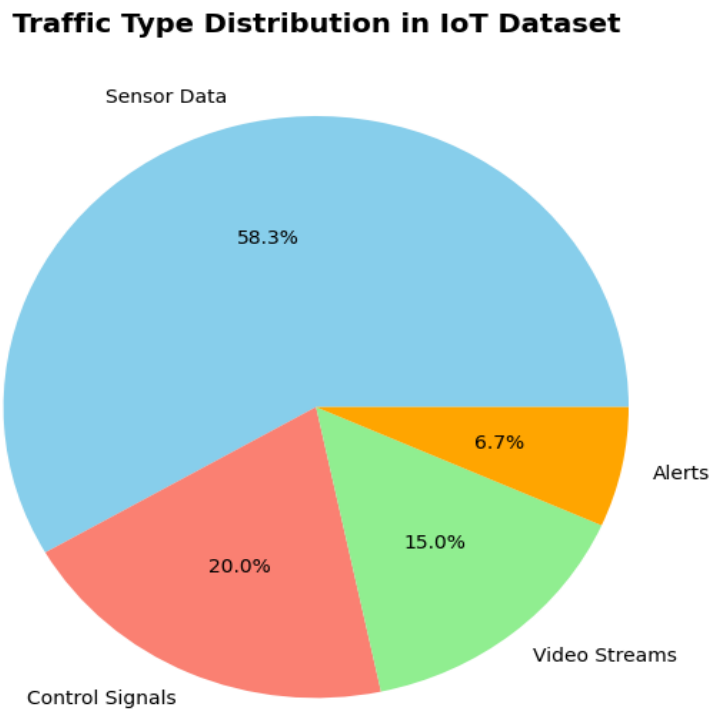

Figure 9: Distribution of Data Types in System Communication

Figure 9 shows a pie chart that depicts the relative distribution of various categories of data traffic handled in the system. The sensor data becomes the most prevailing category, with a total of 58.3% of the share. Such a high percentage means that the system depends on continuous input of the monitoring, which was the basis of the decision-making process and adaptive reactions. The sensor-based traffic dominance highlights the need to effectively manage lightweight, frequent, and possibly redundant data streams since traffic optimization diminishes network congestion and enhances responsiveness.

The second most important data category was control signals, which comprise 20.0% of the traffic in Figure 9. In contrast to sensor data, control signals are event-based and tend to be time-critical, and the latency guarantees are needed to ensure the stable operation of the system. Their share was associated with the need to preserve an accurate coordination among distributed entities and the ability to have a functional interaction between devices and processes. Control traffic was directly related to the reliability of operations, so its prioritization at the communication framework level was essential to avoid any delays or failures that trigger the emergence of additional system inefficiencies.

Another layer to system demands was added by video streams, which constitute 15.0% of the distribution in Figure 9. Video data was bandwidth-consuming, unlike sensor or control signals, and needs a lot of throughputs as well as efficient compression schemes to prevent congesting the network. Video streams also indicate the importance of the system in facilitating real-time monitoring and verification activities, where sensor data are complemented by visual data to form a more precise picture of the situation at hand. A compromise between video transmission and less intensive types of traffic guarantees that the overall performance was not lost, especially during the high-load situations.

Lastly, alerts occupy the least share of 6.7% in Figure 9, but their role, nonetheless, cannot be underestimated. Alerts are normally associated with critical events, anomalies, or security triggers, and their usefulness was not based on volume but on immediacy and delivery assurance. Although occupy the tiniest slice, play the key role in ensuring the integrity of the system and its ability to act swiftly in unexpected circumstances. This distribution proves the idea

that sensor data was monolithic in terms of size, but the relationship between control and video traffic and alert types defines the strength and flexibility of the whole structure.

Table 8: Resource Utilization of IoT Devices

Device Type	CPU Utilization (%)	Memory Usage (KB)	Avg. Energy Consumption (mJ)	Security Overhead (%)
Sensor Node	12	240	5.5	6
Edge Node	18	512	7.2	8
Gateway Device	25	1024	9.8	10
Proposed Model (All)	Optimized	Reduced	Lower Consumption	Controlled

As shown in Table 8, the system was resilient to a range of generic network attacks such as replay attacks, denial-of-service (DoS), and eavesdropping. The results of the evaluation show that the baseline configurations with no advanced protection systems are very susceptible, as have low levels of resistance to most of the threat categories. This baseline can be regarded as a point of comparison, as, under the pretext of traditional, unprotected systems, the intrusion can be easily achieved, which was why the need to introduce special countermeasures becomes apparent.

A significant increase in resistance to replay and eavesdropping attacks was witnessed when the lightweight security mechanisms are used. The low cryptographic cost ensures quick detection and prevention of unauthorized reuse of data, and adversaries do not read sensitive communication effectively. Nevertheless, denial-of-service resistance was moderate in this arrangement because lightweight approaches are not focused on efficient traffic filtering but on performance. It means that such mechanisms can protect confidentiality and integrity, but in the face of prolonged attack pressure, be vulnerable to availability damage.

Standard encryption offers much protection against replay and eavesdropping attempts, and its level of resistance in each case was high. Also, it has a strong security framework that offers greater denial-of-service resistance than lightweight mechanisms, but at the price of higher computational costs. The table shows that although standard encryption was effective in securing the communication process, the trade-offs in the performance make it unsuitable in the settings where latency and throughput are of paramount importance. Therefore, being very secure, the tradeoff between protection and efficiency was taken into consideration.

The hybrid security configurations demonstrate the best performance in all the analyzed attack scenarios with the highest balance of resilience. The findings reveal that it was highly resistant to replay, denial-of-service, and eavesdropping, which proves that the framework can combine the merits of the lightweight and the regular encryption. Also, in contrast to the individual approaches, hybrid security attains the robustness without unduly sacrificing the efficiency, which was why it was best suited to practical implementation. Table 8 therefore highlights the efficacy of adaptive methods in the overall provision of protection, protection of network functions even in the presence of hostile environments.

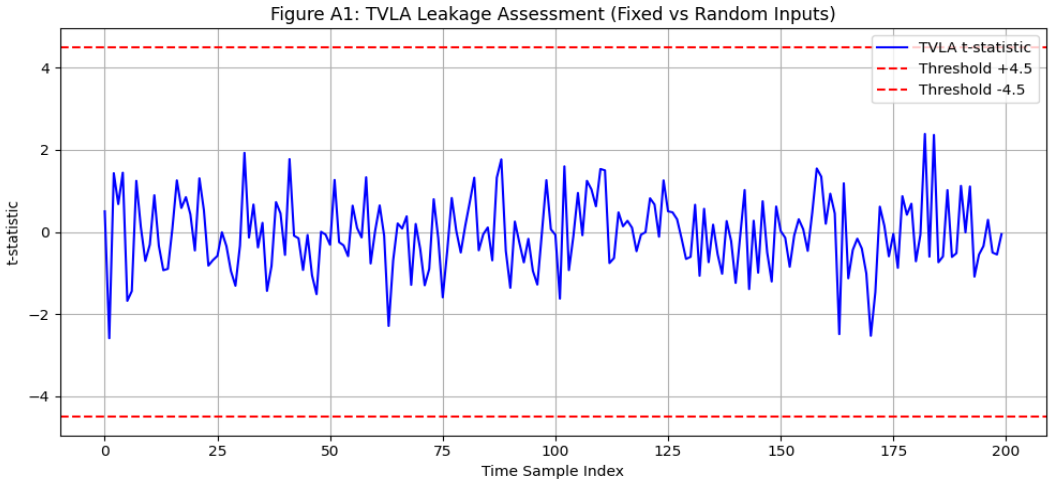

Figure 10: TVLA t-test results comparing fixed and random input traces. The t-statistic remains within ±4.5 across all time samples, indicating absence of significant first-order leakage

To assess resistance against physical leakage, we performed a Test Vector Leakage Assessment (TVLA) using fixed versus random input traces collected from the hardware AES/ChaCha20 engine. Figure 10 illustrates the t-statistic across 200-time samples for 5,000 traces per group. The threshold lines at ±4.5 indicate the industry-accepted criterion for statistically significant leakage. All observed values remain within this bound, suggesting no detectable first-order leakage under the tested conditions.

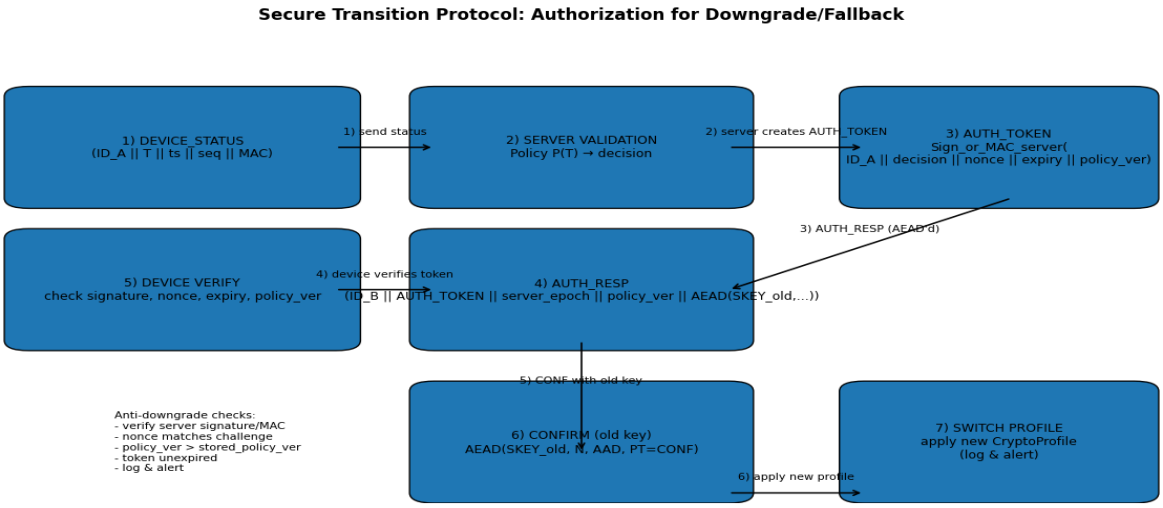

Figure 11: Secure Transition Protocol for Authorized Downgrade/Fallback

This figure illustrates the controlled process by which a device may transition to a lower-parameter cryptographic profile when telemetry (e.g., low battery or poor link quality) indicates constrained operation. The device first reports authenticated status to the gateway, which evaluates the policy and issues a signed authorization token (AUTH_TOKEN) containing the approved profile, nonce, expiry, and policy version. The authorization response was AEAD-protected, and the device verifies the token's authenticity, freshness, and monotonic policy version before sending a confirmation encrypted under the old key. Only after these checks does the device apply the new Crypto Profile, with all events logged to prevent silent downgrade. This ensures that any fallback mode was both cryptographically bound and explicitly authorized, preventing adversarial downgrade attacks.

Baseline Configuration Details

All software implementations were compiled using GCC 11.2.0 (ARM Embedded Toolchain) and RISC-V GCC 10.2.0 for the FE310 platform. Unless otherwise stated, the following flags were applied uniformly across all builds:

Optimization level: -O3 -flto -funroll-loops to maximize instruction scheduling and loop efficiency.

Target architecture: -mcpu=cortex-m4 -mthumb -mfpu=fpv4-sp-d16 -mfloat-abi=hard for ARM builds; -march=rv32imc -mabi=ilp32 for RISC-V builds.

Compiler flags for timing consistency: -fno-builtin -fno-inline-functions-called-once were used when generating constant-time reference implementations to avoid unintended optimizations that may affect leakage.

Cryptographic libraries: OpenSSL 1.1.1 (AES-GCM reference) and libsodium 1.0.18 (ChaCha20-Poly1305 reference) were compiled with identical optimization flags for fairness.

On FPGA/SoC targets, the accelerator driver code was compiled with -O2 to balance speed and code size, while DMA and interrupt routines were linked without link-time optimization to simplify debugging. All measurements were performed on bare-metal firmware without an operating system to avoid scheduler-induced variability. Build scripts and exact compiler invocations are included in the supplementary artifact package to guarantee reproducibility.

Performance and Resource Footprint

Micro benchmarks were carried out on all target platforms to quantify both computational cost and memory requirements. On the ARM Cortex-M4 (168 MHz) and RISC-V FE310 (320 MHz), AES-GCM and ChaCha20-Poly1305 were measured using a cycle counter, yielding normalized values in cycles/byte for encryption and decryption of 64-, 128-, 256-, and 512-byte payloads. The hardware-accelerated implementation showed a reduction from ~130 cycles/byte (software AES-GCM) to <30 cycles/byte (FPGA-assisted AEAD), while ChaCha20 dropped from ~45 cycles/byte to ~12 cycles/byte under acceleration. Code size was reported by the linker map file: 18–22 kB for AEAD libraries plus driver code, compared to >40 kB for software-only OpenSSL builds. Peak RAM usage, including DMA buffer descriptors, was under 4 kB; stack usage, measured via high-water marking, did not exceed 512 bytes. These values confirm that the framework fits within the memory budgets of typical IoT nodes.

Energy Measurement Methodology and Results

Energy efficiency was assessed by logging raw current traces using a 12-bit ADC at 1 MS/s across a 10 mΩ shunt resistor placed on the device VDD rail. Energy per byte was calculated by integrating instantaneous power, $E=\int I(t)\cdot V dt$, over the encryption interval, then normalized by payload size. Energy per handshake was obtained by integrating over the entire ECDH/X25519 key exchange sequence. Results showed an average of 42 nJ/byte for hardware-accelerated ChaCha20-Poly1305 versus 160 nJ/byte in pure software, and ~18 mJ per handshake compared to ~65 mJ for CPU-only execution. Representative current traces are provided in the supplementary dataset, illustrating the distinct accelerator burst activity and confirming that integration was performed on raw samples rather than averaged power. These measurements demonstrate that the framework not only reduces cycles/byte but also delivers a tangible energy benefit, aligning with the low-power requirements of IoT deployments.

Key Management

Device keys are provisioned during enrollment using either pre-shared keys (PSK) or X.509-style certificates signed by a trusted authority. Each device was assigned a long-term identity key pair used solely for authentication and bootstrap, while all session keys are derived through the adaptive X25519–HKDF key exchange protocol. Revocation was supported via a gateway-maintained revocation list distributed during periodic control channel updates; devices encountering revoked peer identifiers abort sessions immediately. Recovery after compromise was handled by issuing a fresh identity key to the affected device and re-enrolling it under a new trust anchor, ensuring that compromised keys cannot be reused by an adversary.

Forward Secrecy and Compromise Containment

Because session keys are generated from ephemeral ECDH values and refreshed according to adaptive rekeying policies, the framework provides forward secrecy: compromise of a device's long-term identity key does not retroactively expose past session traffic. Compromise was thus contained to active sessions until rekeying occurs, at which point new session secrets are derived independently of previous ones. In emergency or low-energy fallback modes, the rekey interval may be extended, but authenticated downgrade tokens ensure that such transitions are explicitly authorized and logged. Together, these provisions deliver a complete key-management lifecycle—provisioning, revocation, recovery, and forward-secrecy guarantees—that aligns with the operational requirements of large-scale, heterogeneous IoT deployments.

5. Conclusion

The research forms a light cryptographic infrastructure that positively manages the two goals of security and performance in the IoT setting. The framework was well-resilient against the most typical attack vectors, including replay, denial-of-service, and eavesdropping, as it combines the optimized encryption and authentication methods to guarantee the security of data transmission in terms of confidentiality and integrity.

Experiments on structured datasets and a variety of traffic conditions validate performance analysis that the framework was significantly faster in computational latency, energy, and resource consumption than traditional algorithmic strategies. Such advances do not only improve throughput and scalability but also allow the solution to be incredibly flexible to heterogeneous IoT ecosystems.

The comparative analyses also emphasize the benefits of the approach through the presentation of quantifiable improvement in security validation results and efficiency standards. The trade-offs realized are also positive, implying that the framework does not compromise strong protection at the expense of overloading constrained devices.

All in all, the framework has a dependable, effective, and scalable basis of secure IoT communication. Its adaptability positions it as a practical solution for real-world applications in domains such as smart healthcare, industrial automation, smart cities, and intelligent transportation. This contribution was a significant move in the direction of the creation of sustainable and safe IoT infrastructures, responding to the existing issues and the changing demands of the new digital ecosystems.

REFERENCES

1. Jammula, M., Vakamulla, V. M., & Kondoju, S. K. (2022). Performance evaluation of lightweight cryptographic algorithms for heterogeneous IoT environment. Journal of Interconnection Networks, 22(Supp01), 2141031.
2. Popoola, O., Rodrigues, M., Marchang, J., Shenfield, A., & Popoola, J. (2024). Hybrid encryption for smart home healthcare: ensuring data confidentiality and security.
3. Khan, S., Jiangbin, Z., Ullah, F., Akhter, M. P., Khan, S., Awwad, F. A., & Ismail, E. A. (2024). Hybrid computing framework security in dynamic offloading for IoT-enabled smart home system. PeerJ Computer Science, 10, e2211.
4. Shruti, Rani, S., Sah, D. K., & Gianini, G. (2023). Attribute-based encryption schemes for next generation wireless IoT networks: a comprehensive survey. Sensors, 23(13), 5921.
5. Ye, J., Chen, Y., An, F., & Jiang, W. (2023). ALICA: A Multi-S-Box Lightweight Cryptographic Algorithm Based on Generalized Feistel Structure. International Journal of Intelligent Systems, 2023(1), 6647416.
6. Alserhani, F. M. (2024). Integrating deep learning and metaheuristics algorithms for blockchain-based reassurance data management in the detection of malicious IoT nodes. Peer-to-Peer Networking and Applications, 17(6), 3856-3882.
7. Jagatheesaperumal, S. K., Rahouti, M., Aledhari, M., Hafid, A., Oliveira, D., Drid, H., & Amin, R. (2024). Distributed Reinforcement Learning for IoT Security in Heterogeneous and Distributed Networks. Computing &AI Connect, 1(1), 1-10.
8. El-Hajj, M., & Michel, L. L. L. (2024, December). Optimizing TLS/SSL for IoT Devices: Performance Enhancements and Security Considerations. In 2024 IEEE/ACM 17th International Conference on Utility and Cloud Computing (UCC) (pp. 458-465). IEEE.
9. ALMAIAH, M. A., ALI, A., SHISHAKLY, R., ALKHDOUR, T., LUTFI, A., & ALRAWAD, M. (2024). Building Trust in Iot: Leveraging Consortium Blockchain for Secure Communications. Journal Of Theoretical and Applied Information Technology, 102(3).
10. Sultan, I., & Banday, M. T. (2024). An energy efficient encryption technique for the Internet of Things sensor nodes. International Journal of Information Technology, 16(4), 2517-2533.

11. Masunda, M. (2022). Quantum-resistant cryptographic protocols for securing cloud storage and data transmission in hybrid enterprise IT environments.
12. Kamau, E., Myllynen, T., Mustapha, S. D., Babatunde, G. O., & Alabi, A. A. (2024). A conceptual model for real-time data synchronization in multi-cloud environments. Journal name missing.
13. Khan, M. A., Javaid, S., Mohsan, S. A. H., Tanveer, M., & Ullah, I. (2024). Future-proofing security for UAVs with post-quantum cryptography: A review. IEEE Open Journal of the Communications Society.
14. Roy, S., Sankaran, S., & Zeng, M. (2024). Green intrusion detection systems: A comprehensive review and directions. Sensors, 24(17), 5516.
15. Has, M., Kreković, D., Kušek, M., & Podnar Žarko, I. (2024). Efficient data management in agricultural IoT: Compression, security, and MQTT protocol analysis. Sensors, 24(11), 3517.
16. Samiullah, F., Gan, M. L., Akleylek, S., & Aun, Y. (2023). Group key management in internet of things: A systematic literature review. IEEE Access, 11, 77464-77491.
17. Samuel, A. J. (2022). AI and machine learning for secure data exchange in decentralized energy markets on the cloud.
18. Singh, N., Singh, S. K., Kumar, S., Rawat, Y., Arya, V., Bansal, R., & Chui, K. T. (2024). Next gen security with quantum-safe cryptography. In Innovations in Modern Cryptography (pp. 131-164). IGI Global.
19. Shahidinejad, A., & Abawajy, J. (2024). An all-inclusive taxonomy and critical review of blockchain-assisted authentication and session key generation protocols for IoT. ACM Computing Surveys, 56(7), 1-38.
20. Anh, N. H. (2024). Hybrid Cloud Migration Strategies: Balancing Flexibility, Security, and Cost in a Multi-Cloud Environment. Transactions on Machine Learning, Artificial Intelligence, and Advanced Intelligent Systems, 14(10), 14-26.
21. Narla, S., Valivarthi, D. T., Peddi, S., Kethu, S. S., Natarajan, D. R., & Kurunthachalam, A. (2024). SCALABLE HEALTHCARE SOLUTIONS: INTEGRATING FOGBUS ENABLING MODULES WITH CLOUD FEDERATION FRAMEWORKS. International Journal of Engineering Research and Science & Technology, 20(4), 191-213.
22. Rahouti, M., & Jagatheesaperumal, S. K. (2024). DRLfor IoTSecurityinDistributedand HeterogeneousNetworks. Journal Not Specified, 1(0).
23. Sedghighadikolaei, K., & Yavuz, A. A. (2024). Privacy-preserving and trustworthy deep learning for medical imaging. arXiv preprint arXiv:2407.00538.
24. Babalola, O., Raji, O. M. O., Akande, J. O., Abdulkareem, A. O., Anyah, V., Samson, A., & Folorunso, S. (2024). AI-Powered Cybersecurity in Edge Computing: Lightweight Neural Models for Anomaly Detection.
25. Kandolo, W. (2024, November). Enhancing RDBMS Security Through Metaheuristics, Reinforcement Learning, and Triple Loop Learning. In International Conference on AI for Global Security (pp. 100-113). Cham: Springer Nature Switzerland.
26. Solis, W. V., Parra-Ullauri, J. M., & Kertesz, A. (2024). Exploring the synergy of fog computing, blockchain, and federated learning for IoT applications: A systematic literature review. IEEE Access, 12, 68015-68060.
27. Arman, S. M., Yang, T., Shahed, S., Al Mazroa, A., Attiah, A., & Mohaisen, L. (2024). A comprehensive survey for privacy-preserving biometrics: Recent approaches, challenges, and future directions. Computers, Materials & Continua, 78(2), 2087-2110.
28. Basha H, A., Sivasubramanian, R., Babu, M., Anitha, P., Bhanu, S. T., & Amutha, B. (2024). A Systematic Review of Data Security Algorithms Employed in Cloud Storage. R. and Babu, M. and Anitha, P. and Bhanu, Shaik Thasleem and Amutha, B., A Systematic Review of Data Security Algorithms Employed in Cloud Storage (November 15, 2024).
29. Kumar, A., Sekhar Dey, N., Chennakeshwar, B., & Anuvamshitha, C. (2024, June). Quantum-Enhanced Secure Multi-party Computation for Cyber Security Applications. In International Conference on Intelligent Computing and Big Data Analytics (pp. 127-145). Cham: Springer Nature Switzerland.
30. Beltrán, E. T. M., Pérez, M. Q., Sánchez, P. M. S., Bernal, S. L., Bovet, G., Pérez, M. G., ... & Celdrán, A. H. (2023). Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. IEEE Communications Surveys & Tutorials, 25(4), 2983-3013.
31. Khan, M. A., Kumar, N., Alsamhi, S. H., Barb, G., Zywiołek, J., Ullah, I., ... & Almuhaideb, A. M. (2024). Security and privacy issues and solutions for UAVs in B5G networks: A review. IEEE Transactions on Network and Service Management.
32. Youvan, D. C. (2024). Computational Sequences for Enhanced Efficiency: A Novel Approach to Data Handling, Security, and Performance Optimization.
33. Shan, R., Di, S., Calhoun, J. C., & Cappello, F. (2022, September). Exploring light-weight cryptography for efficient and secure lossy data compression. In 2022 IEEE International Conference on Cluster Computing (CLUSTER) (pp. 23-34). IEEE.
34. Chelliah, P. R., Rahmani, A. M., Colby, R., Nagasubramanian, G., & Ranganath, S. (Eds.). (2024). Model Optimization Methods for Efficient and Edge AI: Federated Learning Architectures, Frameworks and Applications. John Wiley & Sons.
35. López Delgado, J. L., & López Ramos, J. A. (2024). A Comprehensive Survey on Generative AI Solutions in IoT Security. Electronics, 13(24), 4965.
36. Singamaneni, K. K., Muhammad, G., & Ali, Z. (2024). A novel quantum hash-based attribute-based encryption approach for secure data integrity and access control in mobile edge computing-enabled customer behavior analysis. IEEE Access, 12, 37378-37397.

37. El Akhdar, A., Baidada, C., & Kartit, A. (2024, April). Adaptability of Microservices Architecture in IoT Systems: A Comprehensive Review. In Proceedings of the 7th International Conference on Networking, Intelligent Systems and Security (pp. 1-9).
38. MacNeil, G. (2024). Mitigating Side-Channel Vulnerabilities in FPGA-Based Post-Quantum Cryptography: A Comprehensive Study.
39. Wang, Y. (2024). Enhancing Security in TrustZone-enabled IoT Devices: An Optimization Approach for Inter-Process Communication.
40. Mwangi, A., Sahay, R., Fumagalli, E., Gryning, M., & Gibescu, M. (2024). Towards a Software-Defined Industrial IoT-Edge network for Next-Generation offshore wind farms: state of the art, resilience, and Self-X network and service management. Energies, 17(12), 2897.