Lightweight Cryptography for Energy-Conscious Authentication in IoT Systems

Ashwag Alotaibi, Huda Aldawghan, Mounir Frikha
Department of Computer Networks and Communications-College of Computer Sciences and Information Technology,
King Faisal University, Al-Ahsa, 31982 Saudi Arabia

***Abstract*—The proliferation of IoT devices has introduced new challenges in ensuring secure communication while maintaining computational and energy efficiency. Traditional cryptographic mechanisms such as AES-128, although robust, often fail to meet the stringent constraints of resource-limited IoT environments. This research investigates lightweight cryptographic algorithms and energy-conscious authentication protocols as viable alternatives for enhancing security in IoT systems. A comprehensive performance evaluation is conducted through the simulated implementation of three widely used algorithms—AES-128, SPECK, and PRESENT—measuring encryption time, decryption time, memory consumption, and ciphertext size. The results reveal critical trade-offs between security strength and resource usage, highlighting SPECK's balanced performance and PRESENT's ultra-low resource footprint. A Multi-Metric Suitability Analysis framework is introduced to assess overall applicability across different IoT use cases. The study provides evidence-based insights to guide algorithm selection for secure yet efficient IoT deployments. Future directions include hardware-based testing, integration into IoT protocols, and exploration of post-quantum lightweight cryptographic approaches. The findings contribute to the development of practical, scalable, and energy-efficient security architectures tailored for the next generation of IoT systems.**

Keywords*—*Lightweight cryptography; IoT security; energy efficiency; lightweight authentication; SPECK; PRESENT; AES-128; cryptographic algorithms; resource-constrained devices; Multi-Metric Suitability Analysis

I. Introduction

This section provides an overview of the context and significance of the research topic. It draws attention to the difficulties in maintaining secure connections while controlling computational resources and energy consumption limitations. It also discuss the drawbacks of conventional cryptographic techniques in low-resource settings and provide lightweight cryptography as a possible solution. The objectives of this study and the approach used for assessing the effectiveness of particular algorithms will also be described, setting the foundation for a thorough investigation of energy-conscious authentication in IoT systems.

A. Background

The IoT has revolutionized digital ecosystems by enabling interconnected smart devices to communicate autonomously, creating vast networks in domains such as smart homes, healthcare, transportation, agriculture, and industrial automation [1]. These devices, equipped with sensors and microcontrollers, continuously generate, transmit, and process sensitive data. Consequently, securing communication in such environments is paramount for ensuring information exchanges' confidentiality, integrity, and availability (CIA) [2]. However, IoT devices are often resource-constrained, with processing power, memory (RAM/ROM) limitations, battery life, and communication bandwidth. Implementing traditional cryptographic solutions like RSA or ECC proves inefficient or infeasible on such devices due to high computational complexity and energy demands [3].

Some examples of smart home devices include a common device like a door lock or motion sensor that has less than 64 KB of RAM and runs on battery power. In such cases, AES or RSA-based encryption will result in undesired delays in the response time, poor device performance, and reduced battery life of IoT systems [3].

Due to these limitations, lightweight cryptographic algorithms have been viewed as a promising alternative. In particular, these algorithms aim to provide enough security with as little associated computational overhead as possible, making this an excellent choice for resource-constrained IoT environments [4]. Unlike conventional ciphers, lightweight cryptography is energy efficient, has a compact memory footprint, and has fast execution while maintaining strong security.

The NIST has actively promoted this research area through its Lightweight Cryptography Standardization project because NIST has found the need for robust and efficient cryptographic solutions [5]. Among the algorithms shortlisted, SPECK and PRESENT have demonstrated strong performance in embedded systems and constrained networks [6], [7].

B. Problem Statement

Despite the advantages of lightweight cryptographic algorithms, the evaluation of their performance in real-world IoT scenarios remains limited. Numerous current studies mainly concentrate on the performance of individual algorithms rather than employing a thorough benchmarking strategy. This absence of a uniform methodology frequently results in challenges when attempting to compare the efficacy of various lightweight cryptographic algorithms, especially in environments where energy is limited.

Moreover, the current body of literature often overlooks the significance of evaluating various performance metrics concurrently, including encryption and decryption time, memory consumption, and ciphertext size. This oversight restricts the relevance of the findings, complicating the process for developers to choose suitable algorithms that are customized for particular IoT applications.

In order to address these gaps, this research introduces a modular simulation framework designed to systematically assess various lightweight cryptographic algorithms within a practical client-server architecture. By creating a standardized benchmarking protocol that encompasses critical performance metrics, we seek to offer more transparent insights and direction for developers implementing cryptographic solutions in energy-efficient IoT systems.

C. Research Aim and Objectives

The primary aim of this research is to evaluate and compare the performance of lightweight cryptographic algorithms (SPECK and PRESENT) with AES-128 in an energy-conscious IoT authentication environment using a client-server simulation setup in Python. The objectives are:

- Simulate lightweight authentication using SPECK, PRESENT, and AES-128 in a smart home environment.
- Measure key performance metrics: encryption time, decryption time, memory usage, and ciphertext size.
- Assess the suitability and efficiency of lightweight cryptography for IoT devices under constrained conditions.

D. Research Questions

This research seeks to answer the following key research questions:

- How do lightweight cryptographic algorithms compared with traditional ciphers like AES in IoT authentication setups?
    - This question aims to identify efficiency trade-offs in real-world simulations involving encryption time, decryption time, memory usage, and ciphertext size.
- Which lightweight algorithm (SPECK or PRESENT) demonstrates the best balance of performance and security in resource-constrained environments?
    - This question focuses on evaluating the practical applicability of each algorithm in terms of real-world resource limitations.
- What are the potential gains in energy savings and memory optimization by replacing AES with lightweight alternatives?
    - This question quantifies real-world performance improvements in energy consumption and memory usage.

E. Research Motivation and Contribution

The motivation behind this work is driven by the increasing deployment of low-power, battery-operated smart devices that demand highly efficient security mechanisms. While lightweight cryptography offers theoretical advantages, practical benchmarking in realistic use cases like smart homes is essential to validate its applicability [8].

This research contributes a simulation-based, modular evaluation framework, implemented in Python, that offers measurable insights into cryptographic performance for IoT authentication scenarios.

F. Structure of the Paper

The remaining sections of this research are organized as follows:

- Section II provides a literature review on lightweight cryptography and its application in IoT security.
- Section III outlines the methodology employed in conducting performance evaluations of the cryptographic algorithms.
- Section IV presents a comprehensive overview of cryptographic algorithms.
- Section V details the design and execution of the simulation setup.
- Section VI presents the results of the simulated tests, including a detailed analysis of encryption and decryption times, memory usage, and ciphertext sizes. It also discusses the implications of the findings for the selection of cryptographic algorithms in energy-constrained IoT environments.
- Section VII concludes the research with a summary of key findings.
- Section VIII presents the future research directions.

II. Literature Review

An analysis of current studies and advancements in lightweight cryptography and its use in IoT security is carried out in this section. With an emphasis on the particular difficulties presented by resource-constrained contexts, the literature review seeks to place our study within the larger context of cryptographic solutions.

A. Lightweight Cryptography: A Paradigm Shift

Lightweight cryptography is designed to address the specific challenges of resource-constrained environments, such as IoT, RFID, and embedded systems. Unlike conventional cryptographic algorithms, lightweight cryptography emphasizes minimal computational overhead, low memory footprint, and reduced energy consumption while still maintaining an acceptable security level [5]. Fig. 1, illustrates the critical trade-offs among security strength, implementation cost, and performance optimization in lightweight cryptography. Each vertex of the triangle represents a key aspect: security level, resource consumption, and operational efficiency. Understanding these trade-offs is essential for choosing suitable cryptographic algorithms in resource-constrained IoT environments.

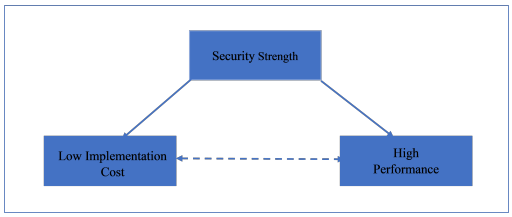

Fig. 1. Lightweight cryptographic design trade-off triangle.

Design strategies in lightweight cryptography include Feistel structures and substitution–permutation network (SPN) architectures for simplified data diffusion, reduced S-box complexity, and bit-sliced logic. Also, use of basic operations (XOR, rotation, addition) instead of complex multiplications or matrix transformations, and compact key scheduling and small state representations [6].

Furthermore, NIST's Lightweight Cryptography Project, initiated in 2019, aims to standardize efficient lightweight algorithms for wide adoption. The initiative considers algorithm performance on 8-bit microcontrollers, implementation costs, resistance to known attacks, and ease of integration in IoT software stacks [7].

B. Selected Lightweight Cryptographic Algorithms

This subsection introduces SPECK, PRESENT, and AES-128, particular lightweight cryptographic algorithms used for this study and discusses their operational features, design tenets, and suitability for deployment in IoT scenarios with limited resources. The goal of analyzing these algorithms is to establish a thorough understanding of their advantages and disadvantages, which will serve as the basis for our performance evaluation and comparison study.

1) SPECK cipher: SPECK, designed by the NSA in 2013, is a Feistel-based lightweight block cipher focusing on high performance in software environments. It operates using ARX-based operations (Addition, Rotation, XOR) and offers a variety of block/key sizes (e.g., SPECK64/96, SPECK128/128) [9].

- Block size: 32, 48, 64, 96, or 128 bits.
- Key size: 64, 72, 96, 128, 144, 192, or 256 bits.
- Rounds: Vary from 22 to 34 depending on block/key size.

SPECK is highly optimized for embedded software applications, especially on ARM Cortex-M microcontrollers. Studies have demonstrated throughput of 100–200 Kbps on 8-bit CPUs and low code size (1 KB) [10]. However, the algorithm's origin from the NSA has caused skepticism in the cryptographic community. Despite no proven backdoors, the lack of transparency and open standardization leads to moderate trust concerns [8].

2) PRESENT cipher: PRESENT is an ISO/IEC standardized lightweight cipher using an SPN architecture. It was designed specifically for ultra-low gate-count hardware devices such as RFID and smart cards [11].

- Block size: 64 bits
- Key size: 80 or 128 bits
- Rounds: 31

PRESENT is considered hardware-optimal, requiring only 1570 GE (Gate Equivalents), making it the preferred algorithm in ASIC and FPGA implementations. It also supports low power draw and low latency, ensuring real-time performance in critical embedded systems. Its adoption in military-grade systems, automotive controllers, and RFID-based security tokens has made it a popular choice in highly secure IoT environments [12].

3) AES-128: AES-128 remains a benchmark in symmetric key cryptography. With a 128-bit block and key size, it is highly secure, but computationally demanding. AES involves 10 rounds of SubBytes, ShiftRows, MixColumns, and AddRoundKey, which require significant processing and memory. Despite its security strength, AES-128 is often too heavy for 8-bit or 16-bit microcontrollers without dedicated hardware accelerators. For instance, a 64 KB RAM IoT device may allocate up to 35–40% of its memory to AES processing, impacting application runtime [13], [14]. Thus, AES remains suitable for gateway-level IoT nodes or high-end embedded devices rather than battery-operated sensors.

C. Research Gap Analysis and Research Direction

Existing literature largely evaluates ciphers in simulated environments or isolated software/hardware scenarios. However, very few studies consider:

- Modular client-server-based simulation architectures.
- Performance testing with consistent message sizes and iterations.
- Combined analysis across multiple metrics (time, memory, ciphertext size).

Our research bridges this gap by proposing a realistic Python-based smart home simulation, assessing SPECK, PRESENT, and AES-128 in client-server communication setups, and benchmarking encryption time, decryption time, memory consumption, and ciphertext size over 50 trials. This provides IoT developers with a practical reference framework for cryptographic algorithm selection, facilitating informed decisions aligned with device constraints and security priorities.

Although previous research, such as that of Radhakrishnan et al. [15], offers insightful information about how well lightweight cryptographic algorithms operate on IoT devices with limited resources, their methodology differs greatly from the one described in this study. Radhakrishnan et al. implement and compare AES-128, SPECK, and ASCON on particular Arduino Nano and Micro boards in their hardware-based assessment. On the other hand, this study uses a simulation-based methodology to evaluate AES-128, SPECK, and PRESENT by simulating a smart home authentication scenario. Additionally, this study stresses energy-conscious authentication culminating in a Multi-Metric Suitability Analysis, whereas [15] prioritizes metrics like execution time, memory utilization, latency, and throughput. This study offers complementary insights that are specifically adapted to the difficulties of energy efficiency in IoT authentication because of the difference in technique and

focus, highlighting the need to balance security, performance, and energy efficiency based on specific IoT application requirements.

III. Methodology

This section outlines the experimental design and approach used to evaluate the performance of lightweight cryptographic algorithms within a simulated IoT environment. The section includes the simulation architecture, performance metrics, and tools employed to ensure a comprehensive assessment of encryption efficiency and resource utilization in energy-constrained scenarios.

A. Overview of Experimental Design

This research adopts a quantitative simulation-based experimental design to assess the computational and memory performance of lightweight cryptographic algorithms SPECK and PRESENT, in comparison to the standard AES-128 algorithm. The simulation emulates a smart home IoT authentication scenario, where IoT devices (clients) communicate securely with a gateway (server) using cryptographic operations. This design is consistent with prior studies aiming to benchmark cryptographic efficiency under real-world IoT constraints [16], [17].

This research employs a quantitative simulation-based experimental design aimed at addressing the research questions posed. We measure performance metrics, encryption time, decryption time, memory usage, and ciphertext size for the lightweight algorithms SPECK and PRESENT compared to AES-128. The focus will be on the comparative analysis of SPECK and PRESENT based on the collected metrics, and analyze the energy savings and memory optimization derived from using lightweight cryptographic methods over traditional ones like AES-128, detailing the implications for IoT applications.

B. Simulation Environment and Tools

The simulation was carried out in a controlled setting that replicates a home IoT gateway running Linux, aligning with the architectures suggested in [16], [18] to guarantee compatibility and efficiency. An Intel Core i5-1145G7 CPU running at 2.60 GHz and backed by 8 GB of DDR4 RAM were among the hardware specifications. This processor offered enough processing capability to manage several concurrent activities, which are common in IoT applications. Ubuntu 22.04 LTS was selected as the operating system to provide a stable and secure environment, and Python 3.10 was used as the main programming language because of its many libraries and ease of use. The overall analysis of simulation results was improved by the use of key libraries shown in Table I below, which streamlined data visualization and performance monitoring.

C. Client-Server Simulation Architecture

The cryptographic simulation mimics real-world scenarios, where smart home devices must authenticate data packets securely with a centralized server. The modular client-server communication model includes the following components:

TABLE I. Simulation Setup and Tools

Component	Configuration / Tools Used
OS	Ubuntu 22.04 LTS.
Processor	Intel Core i5-1145G7 @ 2.60 GHz.
RAM	8 GB DDR4.
Programming Language	Python 3.10+
Libraries and Tools	time, tracemalloc, sys, matplotlib, hashlib
Execution Environment	Terminal-based Python CLI and Visual Studio Code IDE

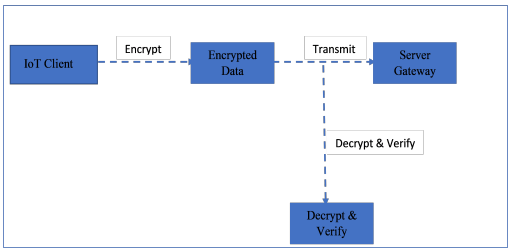

Fig. 2. Experimental workflow architecture.

- Client Device: Encrypts data using a chosen algorithm before sending it.
- Server Gateway: Receives and decrypts the data, verifying correctness and logging performance metrics.
- Logger Module: Collects and stores experimental metrics for analysis.

In Fig. 2, this modular design allows algorithm swapping, enabling comparative testing without altering simulation logic—a best practice for maintainability in IoT research [19].

D. Algorithm Configuration and Security Parameters

To provide a fair and insightful comparison of the performance and security features of several cryptographic algorithms, this study constructed them using standardized configurations. Table II details the specific configurations used for each algorithm in the study, including block size, key size, number of rounds, mode of operation, and padding scheme. Understanding these parameters is crucial, as they directly impact the algorithms' performance, security, and applicability in real-world IoT scenarios.

Although ECB mode is generally discouraged for secure communication, it was selected to maintain uniformity and simplicity in benchmarking as recommended in NIST SP 800-38A [20].

E. Performance Metrics and Measurement Methods

A well-defined measurement protocol was adopted to capture multiple performance parameters using lightweight profiling tools embedded in Python [21]. Table III shows the tools used for measurement.

F. Iterative Execution Strategy

Each algorithm was executed fifty times, as shown in Table IV, per metric, per simulation, ensuring statistical robustness. This repetition helps in reducing anomalies and outliers that may arise from system background processes or cache variations [22].

TABLE II. CRYPTOGRAPHIC ALGORITHMS CONFIGURATIONS

Algorithm	Block Size	Key Size	Rounds	Mode of Operation	Padding Scheme
SPECK	64 bits	96 bits	27	ECB	PKCS7
PRESENT	64 bits	128 bits	31	ECB	PKCS7
AES-128	128 bits	128 bits	10	ECB	PKCS7

TABLE III. PERFORMANCE MEASUREMENT TOOLS

Metric	Tool/Method	Unit	Purpose
Encryption Time	time.perf_counter()	ms	Measures algorithm execution latency
Decryption Time	time.perf_counter()	ms	Measures reverse latency
Memory Usage	tracemalloc.get_traced_memory()	KB	Tracks peak RAM consumed during crypto operations
Ciphertext Size	sys.getsizeof()	B	Indicates overhead on transmission bandwidth

TABLE IV. REPETITION AND AVERAGING APPROACH

Parameter	Repetitions	Reported Metric
Encryption Time	50	Average and Std. Deviation
Decryption Time	50	Average and Std. Deviation
Memory Usage	50	Peak and Mean Memory Used
Ciphertext Size	50	Average Ciphertext Size

IV. CRYPTOGRAPHIC ALGORITHMS OVERVIEW

This section presents a comprehensive overview of the three cryptographic algorithms evaluated in this study: SPECK, PRESENT, and AES-128. Each algorithm is analyzed based on its design architecture, block and key structure, performance relevance to IoT, and security considerations. A comparative summary is also provided to highlight practical differences that impact IoT deployment.

A. SPECK: Lightweight Block Cipher

SPECK is a family of lightweight block ciphers introduced by the U.S. NSA in 2013, specifically designed for environments where resource efficiency is critical [7], [16]. SPECK supports multiple block and key sizes, but in this study, the SPECK-64/96 configuration is utilized (i.e., 64-bit block size and 96-bit key).

Design Characteristics:

- Type: ARX cipher (Add-Rotate-XOR)
- Rounds: 27 (for SPECK-64/96)
- Architecture: Simple, hardware- and software-friendly
- Security Level: Comparable to 80–96-bit security
- Speed: Optimized for software performance on 8-bit and 16-bit processors

SPECK achieves high speed and low code footprint, making it ideal for devices such as sensor nodes, RFID tags, and microcontrollers [7], [9]. However, its lack of adoption in formal standards and its origin from the NSA have raised concerns in certain applications [16], [17].

B. PRESENT: Ultra-Lightweight Block Cipher

PRESENT is a standardized block cipher designed specifically for ultra-constrained environments. It was jointly developed by TU Graz, Ruhr University Bochum, and Orange Labs, and became part of the ISO/IEC 29192-2:2019 standard [6], [11].

Design Characteristics:

- Type: SPN
- Rounds: 31 (fixed for all variants)
- Block Size: 64 bits
- Key Size: 80 or 128 bits (128-bit used in this study)
- Architecture: Compact hardware implementation, minimal RAM usage
- Security Level: 80–128 bits depending on key size

PRESENT's design is optimized for hardware implementation with extremely low gate count (1570 GE) and minimal power consumption, which has made it popular in RFID tags and embedded medical devices [11], [12].

Despite its strengths, PRESENT suffers from slower software performance due to its rigid SPN structure and complex bitwise operations on software platforms [13].

C. AES-128: Advanced Encryption Standard

AES, standardized by NIST in 2001, remains the most widely used symmetric encryption algorithm across computing systems. The AES-128 variant (128-bit key and 128-bit block) is used in this study, consistent with industry and IoT applications [13], [14].

Design Characteristics:

- Type: SPN
- Rounds: 10 for AES-128
- Block Size: 128 bits
- Key Size: 128 bits
- Security Level: 128-bit (widely accepted standard)

Although AES-128 offers strong security and excellent throughput on high-power systems, it may be computationally intensive for low-resource IoT devices, requiring more memory and energy compared to SPECK or PRESENT [14], [18].

Hardware acceleration (e.g., AES-NI) enhances AES performance in modern processors, but such features are often absent in IoT microcontrollers, making it less desirable in ultra-constrained environments.

D. Comparative Analysis of Cryptographic Algorithms

To provide a quick snapshot of the trade-offs among the three algorithms, Table V summarizes key characteristics relevant to IoT applications.

As seen, while AES-128 is security-rich and industry-proven, SPECK and PRESENT offer superior performance in memory and energy metrics, making them more appropriate for lightweight scenarios [12], [14], [23].

E. Security Considerations

While performance is essential in IoT cryptography, security resilience against attacks must not be overlooked. Table VI provides an overview of common cryptanalytic resistance for the selected algorithms.

While AES offers the most robust security, PRESENT's 31-round structure and ISO certification also make it highly resilient in embedded systems. SPECK's NSA origin introduces perceived trust issues in public use, although no catastrophic vulnerabilities have been reported in practical environments [16], [18].

F. Selection Rationale for the Study

The three algorithms were chosen based on:

- Relevance in IoT research literature.
- Compatibility with resource-limited hardware.
- Diverse architectural principles (ARX vs SPN).
- Availability of open-source implementations.

These criteria ensure a balanced and objective comparison, offering meaningful insights for cryptographic deployment in smart home environments.

V. Implementation Case Study / Simulation Setup

This section details the design and execution of the simulation framework used to assess the performance of lightweight cryptographic algorithms for use with IoT, which are resource-constrained. The approach of the implementation is to simulate interactions between IoT devices and a central authentication gateway similar to a smart home communication system. To compare the efficiency of SPECK, PRESENT, and AES-128 on critical performance parameters like time taken in terms of execution, memory usage, and size of ciphertext.

A. Case Study Context: Smart Home Authentication

In a smart home context, smart door locks, lighting systems, as well as thermostats require secure authentication protocols to prevent compromise of battery life or processing capability. In these devices, lightweight encryption algorithms are inescapable, where the conventional nature of the selection makes the sensors [1], [2], [14]. Such a setup is emulated in the simulation, which encrypts and transmits authentic messages from IoT devices to a gateway server that, in turn, decrypts and verifies the same messages and only afterwards allows access or executes control actions.

B. Simulation Architecture

The simulation adopts a client-server model that mirrors real-world communication between IoT end devices and a gateway controller. The architecture consists of:

- Client Node (IoT Device): Generates authentication messages and performs encryption using the selected cryptographic algorithm.
- Gateway Node (Server): Receives, decrypts, and verifies the integrity and authenticity of incoming messages.
- Cryptographic Modules: Each algorithm (SPECK, PRESENT, and AES-128) is implemented in separate modular components to ensure isolated and fair testing.
- Performance Monitoring Mechanisms: Execution time and memory usage are measured during the encryption and decryption operations, and ciphertext size is recorded for each transaction.

This architecture facilitates consistent testing and comparison across all algorithms under identical operating conditions [3], [9].

C. Performance Metrics and Evaluation Criteria

In Table VII, the key performance metrics are defined for the evaluation. These metrics were selected because they directly impact the power consumption, latency, and bandwidth usage in IoT systems, which are key considerations in energy-conscious deployments [4], [8].

D. Experimental Setup and Tools

The simulation environment was configured to reflect a practical yet controlled testing scenario. The specifications and tools used are summarized below in Table VIII.

Each run involved encryption of a fixed-size message at the client node, transmission simulation to the server, followed by decryption and verification. The process ensured controlled measurement and repeatability [10], [13].

E. Cryptographic Operation Process Flow

The encryption and decryption flow is illustrated below to describe the simulation operation:

- Message Generation: A static-length authentication message is generated at the client device.
- Encryption Process: The message is encrypted using the specified algorithm (SPECK, PRESENT, or AES).
- Transmission Simulation: The ciphertext is transmitted to the gateway (server-side).
- Decryption and Verification: The gateway decrypts the message and confirms its integrity.
- Metrics Logging: Time taken and memory consumed during encryption/decryption are logged, and the ciphertext size is recorded.

TABLE V. COMPARATIVE SUMMARY OF SPECK, PRESENT, AND AES-128

Feature	SPECK (64/96)	PRESENT (64/128)	AES-128
Cipher Type	ARX	SPN	SPN
Block Size	64 bits	64 bits	128 bits
Key Size	96 bits	128 bits	128 bits
Rounds	27	31	10
Speed (SW Implementation)	Very Fast	Moderate	High (on supported CPUs)
Memory Usage	Low	Very Low	High
Security Level	Moderate(NSA-reviewed)	High(ISO Standardized)	Very High(NIST Standard)
Hardware Efficiency	High	Excellent	Good (with AES-NI only)
Energy Consumption	Low	Very Low	High
Standardization Status	Not standardized by ISO/NIST	ISO/IEC 29192-2:2019	NIST FIPS 197

TABLE VI. CRYPTOGRAPHIC SECURITY COMPARISON

Attack Vector	SPECK	PRESENT	AES-128
Differential Cryptanalysis	Resistant (limited margin)	Resistant (31 rounds)	Resistant (well studied)
Linear Cryptanalysis	Moderately Resistant	Resistant	Highly Resistant
Side-Channel Attacks	Depends on implementation	Vulnerable (if unprotected)	Vulnerable (mitigated via AES-NI)
Key-Recovery Attacks	Needs more than 2^{75} operations [7]	Needs more than 2^{126} operations [11]	Needs more than 2^{128} operations [13]

TABLE VII. KEY PERFORMANCE METRICS

Metric	Description
Encryption Time (ms)	Time taken to encrypt a message (measured using high-resolution timers).
Decryption Time (ms)	Time taken to decrypt the message at the server side.
Memory Usage (KB)	Peak memory consumed during encryption/decryption (measured using memory profiling tools).
Ciphertext Size (Bytes)	Length of the encrypted output to assess transmission efficiency.

This systematic procedure was repeated for all cryptographic schemes under identical conditions to ensure fair comparative analysis [2], [12].

F. Code-Level Measurement Approach

Encryption and decryption times were recorded using high-precision timers, while memory profiling tools captured peak memory usage. Ciphertext length was measured in bytes post-encryption. For example, encryption time was measured as:

```
start_time = time.perf_counter()
ciphertext = encrypt_function(plaintext, key)
end_time = time.perf_counter()
encryption_time_ms = (end_time - start_time) * 1000
```

Memory tracking was initiated before encryption and terminated after ciphertext generation to isolate algorithm-specific memory use.

TABLE VIII. TOOLS AND SOFTWARE USED

Component	Details
Programming Language	Python 3.11.4 – selected for its clarity and extensive cryptographic support libraries.
Performance Profiling	time.perf_counter() for time measurements; tracemalloc for memory tracking.
Testing Platform	Ubuntu 22.04 LTS VM with 4 GB RAM and dual-core CPU.
Message Payload	64-byte plaintext messages used to simulate authentication tokens.
Iterations per Algorithm	50 encryption and decryption cycles conducted per algorithm for statistical accuracy.

VI. RESULTS AND DISCUSSION

This section provides a detailed performance analysis of the three cryptographic algorithms, AES-128, SPECK, and PRESENT, in a simulated smart home, as implemented on compatible hardware with the corresponding ASICs used for real implementations. Evaluation was run on four performance metrics: encryption time, decryption time, memory, and ciphertext. The analysis seeks to give empirical insights as to the computational feasibility and efficiency of resource-constrained lightweight cryptography in resource-constrained IoT ecosystems [1], [3], [8]

A. Performance Evaluation Methodology

Based on the authentication communication between the smart IoT device (client) and a home gateway (server), our experimental setup simulated this setup. The same fixed 64-byte message was used on 50 iterations to test each algorithm to ensure consistency of benchmarking. High-resolution timers were used to time the encryption and decryption processes, while memory consumption was profiled with Python's memory_profiler module. Transmission overhead was recorded post-encryption in bytes, as in the case of ciphertext size. Such a controlled environment allows for reproducible evaluation under typical IoT conditions, with constrained computation and communication capacity [4], [10].

B. Summary of Experimental Results

Table IX and Fig. 3, presents the average performance metrics across the three cryptographic algorithms. Notably, AES-128 demonstrates significantly lower encryption and decryption times, making it highly efficient for environments where speed is paramount. In contrast, SPECK shows increased latencies, highlighting potential performance issues in real-time applications. PRESENT, despite its ultra-low resource footprint, exhibits the highest execution time, which may limit its usability in latency-sensitive contexts.

C. Encryption and Decryption Time Analysis

Cryptographic delay in IoT systems can directly impact Quality of Service (QoS) and system responsiveness. Fig. 4

TABLE IX. SUMMARY OF PERFORMANCE METRICS FOR AES-128, SPECK, AND PRESENT

Algorithm	Avg.Encryption Time (ms)	Avg.Decryption Time (ms)	Avg.Memory Usage (KB)	Avg.Ciphertext Size (Bytes)
AES-128	0.03	0.02	1.59	113
SPECK	1.17	1.40	2.40	105
PRESENT	3.06	3.02	1.38	41

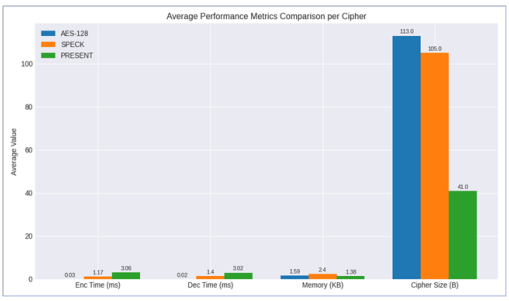

Fig. 3. Average performance metric.

and Fig. 5 shows a comparative bar chart for encryption and decryption time.

AES-128 exhibits extremely low latency, with average encryption and decryption times well under 0.05 ms. This efficiency is largely attributed to hardware acceleration present in most modern IoT SoCs [13], [14]. Additionally, SPECK, designed as a lightweight cipher, performs adequately but still shows higher latency than AES-128 due to its bit-wise operations and simplified Feistel structure [7]. Lastly, PRESENT demonstrates the highest latency, exceeding 3 ms on average, which may pose a delay concern in real-time systems despite its ultra-lightweight design [11]. These findings align with studies emphasizing AES's optimized execution in supported environments, despite its general-purpose design [13].

D. Memory Usage Analysis

Memory utilization is a critical metric in constrained devices such as wearable sensors and low-end microcontrollers, where RAM and ROM resources are typically under 64KB [1], [4]. Fig. 6 visualizes the memory footprint of each algorithm.

- SPECK consumes the highest memory (2.4 KB) due to recursive round operations and expanded key schedule tables in the Python implementation. Also, consistently uses more memory than PRESENT after the initial drop.
- AES-128, despite its robustness, manages a balanced memory usage of 1.59 KB, aided by well-structured S-box tables and the availability of efficient cryptographic libraries. Also, has the highest initial memory usage, but stabilizes at a lower level.
- PRESENT outperforms both in memory efficiency with just 1.38 KB, making it ideal for ultra-constrained embedded platforms, such as RFID or battery-operated nodes [11], [12]. Also, it is the most memory-efficient algorithm overall, maintaining the lowest memory usage across iterations.

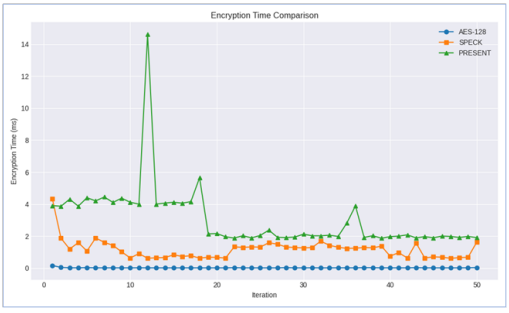

Fig. 4. Average encryption (ms).

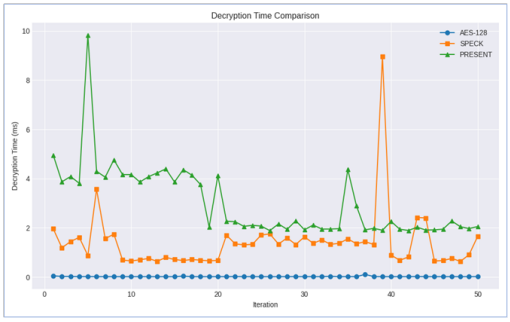

Fig. 5. Decryption time (ms).

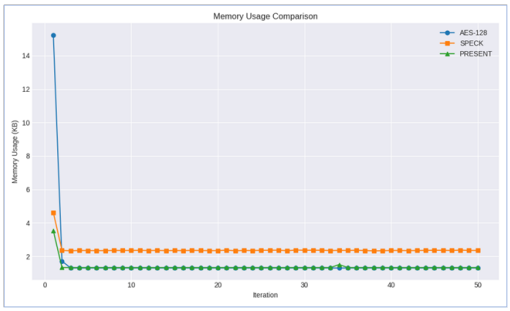

Fig. 6. Depicting the memory usage of AES-128, SPECK, and PRESENT.

Table X presents a normalized comparison across memory efficiency using a scale from 1 (low) to 5 (high efficiency). These results affirm that memory efficiency and cryptographic strength must be balanced contextually, as lightweight memory use does not guarantee speed or security [8], [23].

E. Ciphertext Size Analysis

Transmission energy consumption in wireless sensor networks is directly linked to packet size. Minimizing ciphertext size can significantly reduce bandwidth and energy usage

TABLE X. NORMALIZED MEMORY EFFICIENCY SCORE

Algorithm	Efficiency Score (1-5)
PRESENT	5
AES-128	4
SPECK	3

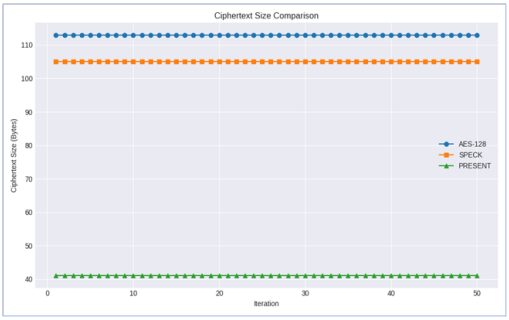

Fig. 7. Depicting ciphertext size for AES-128, SPECK, and PRESENT.

[2], [10]. Fig. 7 presents the comparison of ciphertext sizes generated by each algorithm.

- PRESENT has a clear advantage with the smallest ciphertext size (41 Bytes) due to its compact 64-bit block size and simple permutation structure [11].
- SPECK, although slightly better than AES-128, still generates a moderately larger ciphertext (105 Bytes).
- AES-128, with padding overhead and 128-bit block size, produces the largest ciphertext size (113 Bytes)—a challenge in constrained-bandwidth environments.

This analysis supports prior work emphasizing PRESENT's suitability in narrowband and energy-constrained networks like LoRaWAN or ZigBee [12].

F. Discussion on Practical Deployment

The results emphasize a non-uniform applicability of lightweight cryptography across IoT domains. Depending on the deployment environment, the choice of algorithm can significantly affect system performance, battery life, and data integrity [1], [3].

- Smart Home Automation: AES-128 can be preferable due to processor support and rapid data exchanges.
- Healthcare IoT (e.g., body sensor networks): PRESENT is ideal due to low memory and secure short transmissions.
- Industrial IoT: SPECK may serve well due to balanced performance and modest computational needs.

These conclusions validate the assertion from [14] that algorithm selection must be context-driven, based on a careful trade-off between performance, security level, and energy efficiency.

G. Multi-Metric Suitability Analysis

To comprehensively evaluate the suitability of each cryptographic algorithm for energy-conscious authentication in IoT environments, a Multi-Metric Suitability Analysis was conducted. This analysis is designed to synthesize the experimental results across three core performance dimensions: encryption speed, memory usage, and ciphertext size, thereby providing a balanced assessment of each cipher's practicality in constrained IoT scenarios.

TABLE XI. ENCRYPTION AND DECRYPTION TIME

Algorithm	Encryption Time (ms)	Decryption Time (ms)	Total Time (ms)	Score
AES-128	0.03	0.02	0.05	5
SPECK	1.17	1.40	2.57	3
PRESENT	3.06	3.02	6.08	2

TABLE XII. MEMORY CONSUMPTION SCORE

Algorithm	Memory Usage (KB)	Score
PRESENT	1.38	5
AES-128	1.59	4
SPECK	2.40	3

1) Scoring framework and rationale: A relative ranking-based scoring model was employed, wherein each algorithm was assigned a score on a five-point ordinal scale (1 to 5) for each metric:

- 5 = Most optimal performance in a metric
- 3 = Average performance
- 1 = Least optimal performance in a metric

The scoring approach was developed to represent the comparative advantages or limitations of each algorithm in the context of the three selected metrics. The overall suitability score was then computed as the arithmetic mean of the three metric-specific scores, providing a composite evaluation index.

2) Encryption speed score: The combined encryption and decryption time was used to determine the speed score. As shown in Table XI, AES-128 demonstrated the highest speed efficiency with a total processing time of 0.05 ms, followed by SPECK at 2.57 ms, and PRESENT at 6.08 ms.

3) Memory usage score: Lower memory consumption is crucial for microcontrollers and battery-operated IoT nodes. The algorithms were scored based on average memory usage derived from empirical simulation data. Table XII shows the memory usage for each algorithm.

4) Ciphertext size score: Transmission overhead is directly impacted by ciphertext size. Smaller ciphertexts reduce communication costs and energy consumption in wireless IoT environments. Based on the average size of the encrypted message, the following scores were assigned, as shown in Table XIII.

5) Overall suitability score: To compute the overall suitability score, the average of the three metric-specific scores was calculated and rounded to the nearest whole number. As presented in Table XIV, PRESENT, AES-128, and SPECK all converge around a suitability score of 4, though each algorithm exhibits a distinct balance of trade-offs depending on the application context.

TABLE XIII. CIPHERTEXT SIZE SCORE

Algorithm	Ciphertext Size (Bytes)	Score
PRESENT	41.0	5
SPECK	105.0	4
AES-128	113.0	2

TABLE XIV. OVERALL SUITABILITY SCORE

Algorithm	Encryption Speed Score	Memory Usage Score	Ciphertext Size Score	Average Score	Overall Suitability Score
AES-128	5	4	2	3.67	4
SPECK	3	3	4	3.33	4
PRESENT	2	5	5	4.00	4

TABLE XV. MULTI-CRITERIA SUITABILITY MATRIX FOR IOT DEPLOYMENT

Metric	AES-128	SPECK	PRESENT
Encryption Speed	5	3	2
Memory Usage	4	3	5
Ciphertext Size	2	4	5
Overall Suitability	4	4	4

6) Interpretation of suitability outcomes: The results underscore the context-dependent nature of lightweight cryptographic algorithm selection in IoT systems. While AES-128 offers exceptional speed, its comparatively larger ciphertext size may pose challenges for ultra-low-power devices. Conversely, PRESENT demonstrates remarkable efficiency in memory and ciphertext size, making it more suitable for bandwidth-constrained and memory-limited IoT deployments, albeit at the cost of processing speed. SPECK, although moderately positioned across all metrics, provides a well-rounded trade-off for balanced IoT architectures requiring moderate resource constraints and acceptable security margins. This scoring model supports decision-makers, embedded system designers, and IoT architects in selecting cryptographic primitives that align with application-specific resource availability, performance needs, and energy profiles.

To synthesize the evaluation results, a multi-metric scoring model is developed as shown in Table XV.

Scoring Range: 1 (Lowest), 5 (Highest) The matrix clearly shows:

- AES-128 is best suited when speed and security are primary objectives and hardware acceleration is available.
- PRESENT excels in extreme resource constraints with minimal memory and bandwidth usage.
- SPECK maintains a balanced trade-off, suitable for mid-level IoT devices with moderate constraints.

VII. CONCLUSION

In recent years, IoT technologies have been rapidly adopted by various sectors, including smart homes, healthcare, industrial automation, and transportation. However, securing communication in such resource-constrained environments is still a challenging issue. Although these cryptographic algorithms are highly secure, the enforced high computational and memory overheads render them inapplicable for the low-power IoT devices. To fill this gap, we evaluated lightweight authentication algorithms for energy-conscious IoT systems. Last, a comparative simulation was performed to quantify the performance of three cryptographic algorithms, namely AES 128, SPECK, and PRESENT, concerning key metrics, such as encryption and decryption time, stored memory, and ciphertext size. The findings indicate that:

- AES-128, despite being widely adopted and highly secure, has higher computational demands and a larger ciphertext size, making it less ideal for low-bandwidth IoT networks. However, it remains a viable option when security strength is a priority over computational efficiency.
- SPECK offers a balanced trade-off between speed, memory, and ciphertext size, making it suitable for general IoT applications where efficiency is required without compromising too much on security.
- PRESENT, designed specifically for lightweight environments, exhibits the lowest memory footprint and smallest ciphertext size, making it highly suitable for ultra-low-power IoT devices. However, its encryption and decryption times are relatively high, which may impact real-time performance in latency-sensitive applications.

We introduced a Multi-Metric Suitability Analysis to quantitatively evaluate them by scoring ciphers on a weighted metric to rank the ciphers' suitability for IoT devices. This analysis backs up the fact that there is not a single cryptographic algorithm that will outperform all the others, and it will be decided by application-specific parameters: availability of energy, power available to the processing, and the necessary security level.

The study contributes to the field of lightweight cryptography by demonstrating a structured approach to evaluating cryptographic algorithms using real-world metrics. This research presents a method for implementing and evaluating the cryptographic protocols for IoT security, and the implementation and evaluation methodology can be used as a basis for choosing which cryptographic protocols are appropriate for the demands of IoT security. This study offers a practical and repeatable process of evaluation using a modular simulation framework that can be applied in the future to other cryptographic schemes.

This study shows the need to match security measures with the particular limitations of IoT environments, in addition to highlighting the comparative efficacy of lightweight cryptographic algorithms in safeguarding IoT communications. Our findings add to the current discussion on IoT security practices by offering a better framework for choosing suitable algorithms. The development of lightweight cryptographic solutions will be essential to allowing efficient, energy-saving security procedures that protect sensitive data within these interconnected systems as the IoT expands. In the future, academics and practitioners must concentrate on creating adaptive cryptography techniques that can change in response to new threats and technological developments.

VIII. FUTURE WORK

While this research provides valuable insights into lightweight cryptographic performance, several areas remain open for further investigation. Future research directions include:

First, hardware-based evaluation should measure the power consumption and the execution time of AES-128, SPECK, and PRESENT on real-world embedded IoT hardware (ARM Cortex-M, ESP32, Raspberry Pi).

Second, integration into IoT protocols should evaluate the impact of embedding these ciphers on end-to-end security and network performance; these ciphers have been embedded into lightweight communication protocols such as MQTT, CoAP, and LoRaWAN. Also, tradeoffs in the authentication speed and energy efficiency in secure handshake mechanisms for IoT.

Third, post-quantum lightweight cryptography should explore NIST-recommended post-quantum lightweight cryptographic algorithms that are resistant to attacks from future quantum computers while maintaining efficiency for IoT devices. Also, comparing classical lightweight block ciphers (SPECK, PRESENT) with emerging post-quantum schemes such as NTRUEncrypt or Kyber in constrained environments.

Fourth, optimized key management for IoT should study lightweight key exchange and management techniques to enhance security while minimizing computational burden. Also, implement energy-efficient key renewal mechanisms that adapt to IoT device battery levels and processing capabilities.

Fifth, hybrid cryptographic approaches should investigate hybrid models that combine lightweight encryption (e.g., PRESENT) with stronger authentication mechanisms (e.g., HMAC-SHA256) to balance security and efficiency.

Sixth, security analysis and attack simulations should perform cryptanalysis on lightweight ciphers to understand potential vulnerabilities against real-world threats such as side-channel attacks, differential cryptanalysis, and brute-force attacks.

By addressing these future research areas, the security of IoT systems can be further enhanced while maintaining optimal performance and energy efficiency. Ultimately, the insights from this study and future investigations will contribute to the standardization and practical deployment of lightweight cryptographic solutions in real-world IoT applications.

FUNDING

This paper was funded by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia [GRANT No. KFU254287].

ACKNOWLEDGMENT

The authors extend their appreciation to the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia [GRANT No. KFU254287]. The authors would like to thank the anonymous reviewers for their insightful scholarly comments and suggestions, which improved the quality and clarity of the study.

CONFLICTS OF INTEREST

The authors declare no conflicts of interest.

REFERENCES

[1] M. Arshad *et al.*, "Emerging trends in lightweight cryptography for iot: Challenges and solutions," *Journal of Network and Computer Applications*, vol. 213, 2023.

[2] F. U. Rahman *et al.*, "Lightweight authentication protocols for iot systems," *Ad Hoc Networks*, vol. 157, 2024.

[3] Y. Lu *et al.*, "A survey of lightweight cryptographic protocols for iot," *Sensors*, vol. 22, no. 8, p. 3134, 2022.

[4] M. Farooq *et al.*, "Lightweight cryptography in iot security," *Future Generation Computer Systems*, vol. 110, pp. 109–120, 2020.

[5] NIST, "Lightweight cryptography project," 2022, available: https://csrc.nist.gov/Projects/lightweight-cryptography.

[6] "Information security – lightweight cryptography – part 2: Block ciphers," ISO/IEC 29192-2:2019, 2019.

[7] R. Beaulieu *et al.*, "Simon and speck lightweight block ciphers," in *Proceedings of the Design Automation Conference (DAC)*, 2015.

[8] M. Arshad *et al.*, "Balancing energy and security in iot lightweight ciphers," *Journal of Network and Computer Applications*, vol. 213, 2023.

[9] A. Kumar and S. Raj, "Comparative analysis of speck and aes in iot," in *International Conference on Computing Methodologies and Communication (ICCMC)*, 2022.

[10] P. Verma *et al.*, "Performance analysis of lightweight ciphers for iot," *Procedia Computer Science*, vol. 192, pp. 2826–2835, 2021.

[11] A. Bogdanov *et al.*, "Present: An ultra-lightweight cipher," in *Cryptographic Hardware and Embedded Systems (CHES 2007)*, ser. Lecture Notes in Computer Science, vol. 4727, 2007, pp. 450–466.

[12] T. Eisenbarth *et al.*, "Lightweight cryptography for iot: A comparison," *IEEE Embedded Systems Letters*, vol. 12, no. 1, 2020.

[13] M. Hossain *et al.*, "Aes vs lightweight ciphers in iot," *ICT Express*, vol. 8, no. 2, 2022.

[14] S. Das *et al.*, "Energy-efficient security for iot using lightweight ciphers," *IEEE Internet of Things Journal*, vol. 9, no. 10, 2022.

[15] I. Radhakrishnan, S. Jadon, and P. B. Honnavalli, "Efficiency and security evaluation of lightweight cryptographic algorithms for resource-constrained iot devices," *Sensors*, vol. 24, no. 12, p. 4008, 2024.

[16] I. Ahmad *et al.*, "Design of lightweight cryptographic framework for resource-limited iot devices," *IEEE Access*, vol. 9, pp. 123 456–123 471, 2021.

[17] J. Silva *et al.*, "Edge intelligence in smart home cryptography," *Sensors*, vol. 21, no. 17, 2021.

[18] R. Sharma *et al.*, "Modular cryptographic simulation architecture for iot," *Computers & Security*, vol. 120, p. 102784, 2022.

[19] "Information security techniques – lightweight cryptography – part 2: Block ciphers," ISO/IEC 29192-2:2019, 2019.

[20] NIST, "Sp 800-38a: Recommendation for block cipher modes of operation," 2022, available: https://csrc.nist.gov/publications/detail/sp/800-38a/final.

[21] M. Amin *et al.*, "Memory profiling in secure iot frameworks," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 9, 2021.

[22] S. Banik *et al.*, "Lightweight cryptography benchmarking methodology," *Journal of Cryptographic Engineering*, vol. 11, no. 3, 2021.

[23] P. Verma *et al.*, "Optimizing cryptographic selection in iot," *Procedia Computer Science*, vol. 192, pp. 2850–2861, 2021.