Ingénierie des Systèmes d'Information
Vol. 31, No. 5, May, 2026, pp. 1711-1724
Journal homepage: http://iieta.org/journals/isi

Benchmarking Lightweight Cryptographic Algorithms for Resource-Constrained IoT Devices: MATLAB Simulation and Preliminary Hardware Validation

Shaymaa Abdulhussein Shnain[1*], Zahraa Modher Nabat[2], May A. Salih[3,4]

[1] Math Department, Basic Education College, University of Babylon, Babylon 51002, Iraq
[2] Registration and Students Affairs, University of Babylon, Babylon 51002, Iraq
[3] Information Technology College, University of Babylon, Babylon 51002, Iraq
[4] Department of Computer Techniques Engineering, Alsafwa University College, Karbala 56001, Iraq

Corresponding Author Email: shaima.almorshedy3@uobabylon.edu.iq

https://doi.org/10.18280/isi.310527

Received: 3 December 2025
Revised: 1 March 2026
Accepted: 15 April 2026
Available online: 31 May 2026

Keywords:
Internet of Things, lightweight cryptography, Matrix Laboratory simulation, Raspberry Pi, Arduino, IoT security

ABSTRACT

There is a challenge in ensuring adequate security for Internet of Things (IoT) devices utilizing cryptographic functions because of the limits imposed on individual devices from a memory, processing, and energy budget perspective. The objective of the research paper is to identify a few of the lightweight cryptographic algorithms and validate their suitability to secure IoT devices by simulating those top algorithms using the Matrix Laboratory (MATLAB) programming environment and experimentally verifying their performance on hardware platforms such as the Raspberry Pi and Arduino. All four symmetric algorithms that were selected (Advanced Encryption Standard (AES), SPECK, ASCON, PRESENT) were tested to determine their performance against the criteria for the algorithms utilizing standardized operational constraints including execution time, memory footprint, energy consumption, throughput, and simulated resistance to side-channel attacks. Additionally, elliptic curve cryptography (ECC) was tested to provide a reference asymmetric algorithm for purposes of comparison and establishing an appropriate baseline when executing key agreement and key exchange operations. The results associated with the performance of these algorithms provide the ability to identify and analyze the various trade-off features associated with those individual algorithms. Specifically, SPECK was found to demonstrate the fastest overall execution time (0.30 ms/128-bit block) and throughput (operations per second) which makes it the best-suited algorithm for use in applications where low latency is required to obtain real-time sensor data. ASCON had the lowest estimated energy consumption (15.0 μJ/operation) and possesses authenticated encryption, consequently, making it the best choice for IoT applications with stringent requirements for security and battery power. Although AES can be the selected baseline algorithm, it has the highest requirements for memory and energy of the algorithms tested. Conversely, although PRESENT has one of the lowest memory footprints, it also demonstrated low throughput performance. Another key observation was that ECC is the most applicable asymmetric solution for performing selective authentication via public/private key agreements; however, it is not a viable alternative to provide ongoing encryption to protect data. Ultimately, there will not be any one optimal lightweight cryptographic algorithm that is optimal for all applications that are associated with IoT. Therefore, the algorithm chosen to secure an IoT device must be based on the unique use case circumstances such as latency, available energy, authentication requirements, and capabilities of the device.

1. INTRODUCTION

The Internet of Things (IoT) has rapidly transitioned from an idea believed to come to fruition in a few years to an actuality that has become the most widely applied technological concept we have available in today's world. IoT networks enable us to compile data, exchange information, and make real-time intelligent decisions by establishing connections among billions of devices including household items, wearable healthcare monitors, self-driving vehicles, and sensors present on factory floors. As the existing market trends indicate, the number of the IoT devices will surpass 30 billion throughout the globe in the next years, and that is the scale of such digitalization. Despite being unmatched in terms of efficiency and convenience, IoT presents an unparalleled security and privacy threat which, unless properly addressed, will undermine the trust and user confidence in its systems and stifle the potential optimization of these system applications [1, 2].

IoT device security is especially a challenge with this many devices and highly diverse environmental conditions on which they are being implemented. IoT devices are typically devices with few processing units, storage, and available power as opposed to other computer platforms like laptops and servers.

These are inevitable limitations and are the direct causes of the discrepancy between the computational capabilities of standard cryptographic algorithms and the platform of IoT hardware. Although Rivest–Shamir–Adleman (RSA), Advanced Encryption Standard (AES), and elliptic curve cryptography (ECC) have been used to secure enterprise and desktop computers, their usage in the IoT devices is likely to lead to power consumption, intolerable delays, or system crashes [3]. This is in opposition to the requirement of customized lightweight cryptosystems which will operate in the IoT environment.

In nature, IoT networks have multifold and dynamic threats. These common vulnerabilities are wireless communication wiretapping, in-transit attacks, physical attacks on the sensor nodes, and denial-of-service (DoS) attacks to shut down the normal operation. The impact of the security compromise can be life-threatening in such high-pressure scenarios as medicine, where an IoT sensor reads vital signs and transmits critical data to a doctor. Equally, in industrial control systems, IoT-based attacks can cause operations to go offline, loss or even safety risks. Thus, not only does it become critical that the implementations of IoT are adequately secured, but is now also socially and publicly engaging [2].

The new recent advances in lightweight cryptography have turned out to be the prophecy in sealing this gap. Symmetric key building blocks: PRESENT, SIMON, SPECK and the recently standardized ASCON family have been brought down to the size of a bare minimum in terms of computational, energy loads and sufficient levels of security [1]. Minimization of the building blocks is considered based on minimized key size, minimal operations and optimised performance on hardware-unfriendly devices. Resource-constrained environments, on the other hand, can be provided with secure key exchange and authentication in light-weight implementations of asymmetric cryptographic techniques, e.g., elliptic curve-based cryptography. Additional attempts by the National Institute of Standards and Technology (NIST) in the area of the standardization of light-weight cryptography also demonstrates how internationally, an adoption of this demand has occurred [3].

This document presents a technical overview of lightweight cryptographic algorithms. In addition, valid experimental performance evaluation has been conducted through simulated results from Matrix Laboratory (MATLAB) software and preliminary hardware validation of these same algorithms on IoT platforms. The overall review will represent a total analytic view of lightweight cryptographic algorithm methods in an IoT device environment with both a theoretical focus on lightweight crypto technical design and a practical focus on lightweight algorithm solutions. In order to provide a coherent causal description of the applicability of lightweight crypto to IoT security solutions, the threats posed by current methods are described as well as the taxonomy of low-resource symmetric and asymmetric crypto algorithms. The latest advances in lightweight algorithm technology are focused upon as well—with an emphasis on the ASCON algorithm suite, which is currently being developed as a potential NIST standard for small-footprint crypto systems. Additionally, practical applications of low-resource systems that use Python programming are described, including conducting the "Strength of Security", "Efficiency", and "Feasibility of Scale" for various project scenarios based on technical implementation of low-resource IoT supported devices. The application of conceptual and experimental evidence in this manner will empower researchers, engineers and end-users with sufficient information to design an IoT infrastructure with true security. Thus, because of the ongoing uncontrolled proliferation of IoT technology, it is apparent that a new and closely monitored implementation of the current crypto process is required. The current existing methodologies are inherently strong, but do not necessarily possess sufficient flexibility to accommodate all of today's low-resource IoT-supported device scenarios. The answer to this dilemma of simultaneous provision for security and efficiency is through the use of lightweight cryptography protocols that are being facilitated by the international standards processes and universally proven, real-world applications. This article claims that both IoT technology and cryptographic research fields must grow together in order for users to benefit from the ability to connect anywhere with the seriousness of losing privacy, security and trust. Unlike theoretical reviews, this report combines both empirical verification of algorithm performance with analytical discussion. While previous lightweight cryptographic literature has thoroughly reviewed various algorithms, there has not been as much study done with respect to empirical comparative testing using simulations and proven empirical hardware. This research adds to current literature by providing:

(1) a structured analytical review of selected lightweight cryptographic algorithms utilized in IoT scenarios,

(2) simulations using the MATLAB environment for comparing the performance of those algorithms under limited processing conditions,

(3) preliminary experimental testing of selected algorithms on Raspberry Pi and Arduino platforms to validate their practicality, and

(4) practical recommendations to select proper encryption methods for IoT systems.

The combination of the reputed literature and the empirical analysis results aims to bridge the gap between the theory and practice of lightweight cryptography as applied to the IoT. To accomplish the challenges of implementing secure encryption in resource-constrained IoT, a combination of literature reviews and empirical evaluations served as the hybrid methodology applied in this research. The research first identified the most common lightweight cryptographic algorithms for application to IoT by conducting a comprehensive literature review of the algorithms available for IoT. The identified algorithms were then evaluated by comparing the execution time, processing load and memory usage of each algorithm to ensure all data was collected for analysis under the same conditions by simulating each algorithm in the MATLAB environment. Finally, to validate the results from the simulations, preliminary hardware experiments were completed on both Raspberry Pi and Arduino platforms. The hardware experiments were designed to validate the feasibility of each algorithm being applied to a real-world IoT device and monitored the practical performance attributes such as processing times and resource utilization. This research successfully combines the results of both simulation and limited empirical hardware testing to provide both research and practice-based assessments of the applicability of lightweight cryptographic algorithms for IoT systems. Overall, this research, through its systematic empirical comparative experimentation of lightweight cryptographic algorithms for application in IoT environments, will serve as a contribution to the field of research as follows:

1) A unified framework using both simulation and empirical

hardware experimentation providing a basis for developing an objective understanding of the performance of lightweight cryptographic algorithms,

2) A reproducible method using the MATLAB environment for benchmarking lightweight cryptographic algorithm performance,

3) An empirical examination of the performance characteristics of the lightweight cryptographic algorithms on actual IoT platforms assisting to close the gap between the theory and application of lightweight cryptography in IoT.

2. LITERATURE REVIEW

In the years leading up to 2014, lightweight cryptography was increasingly studied in connection with the increasing deployment of IoT and a requirement for communication protocols that are both light and secure. The overwhelming majority of IoT devices will have far less energy, memory, and computational capability than traditional computer systems, and will therefore need new cryptographic primitives designed for these limitations. The history of lightweight cryptography shows considerable activity in the design, testing, and standardization of sufficient security of these primitives so that they may provide useful services with limited resources in a resource-constrained environment. This review provides a high level summary of the works referenced above, categorized by type of lightweight cryptographic algorithm, the device platforms on which the algorithms have been evaluated, and research gaps for future work.

2.1 Types of lightweight cryptographic algorithms

Lightweight cryptography is typically categorized into block ciphers, stream ciphers, and hash functions, each with varying strengths and compromises towards IoT implementation.

Block ciphers such as PRESENT, SIMON, and SPECK have been of immense interest because of their simplicity and reduced computation requirements. PRESENT, suggested as among the first lightweight block ciphers, has been extensively tested for embedded systems and RFID devices due to its shorter key size and efficient hardware implementation [4]. Similarly, the SPECK block cipher family is extremely software-efficient, particularly on low-power microcontrollers, although its usage has been in question due to skepticism regarding its discovery as well as reported vulnerabilities.

Stream ciphers like Trivium and Grain-128 provide lightweight solutions to encrypt streams of uninterrupted data. Their construction from shift registers makes it possible for one to derive low hardware size, hence being very well-adopted in power-restricted as well as memory-limited environments [5]. Stream ciphers are used on a very large scale in IoT applications dealing with real-time data security, i.e., sensor-gateway communication in wireless sensor networks.

Light hash functions are another critical component of IoT security. Hash functions such as SPONGENT, PHOTON, and Quark are designed to achieve cryptographic properties such as collision resistance and preimage resistance on the utilization of much less resources than traditional algorithms such as SHA-256 [4]. Hash functions form a critical component of IoT setups for authentication, data integrity verification, and secure key derivation.

2.2 Standardization efforts and algorithm benchmarks

The research community in cryptocurrencies acknowledges the effort of standardizing light-weight algorithms for implementation globally. One of the key milestones was achieved when the NIST announced ASCON as the winner of its light-weight standardization program for cryptography [1]. ASCON, initially introduced in the CAESAR competition for authenticated encryption, showed excellent performance in software and hardware and excellent resistance to known cryptanalytic attacks. Standardization serves as a testament to the growing maturity of lightweight cryptography as an independent research field.

Even with the emergence of lightweight special primitives, the AES remains an omnipresent standard. While AES is not the best for ultra-constrained systems, when used in an optimized manner, it can be usable on particular IoT devices with acceptable trade-offs [3]. However, due to the relatively higher computational and memory requirements of AES, its use is practically prohibited on ultra-constrained devices, supporting further the use of others such as ASCON, PRESENT, and Grain-128.

2.3 Performance testing of Internet of Things devices

Several works empirically evaluated light-weight crypto algorithms on various IoT hardware platforms such as Arduino, ESP32, and Raspberry Pi. They typically measure execution time, memory consumption, and energy efficiency, giving a realistic gauge of algorithm suitability [6]. For example, block ciphers such as PRESENT and SPECK are always more energy-efficient on low-end microcontrollers compared to AES. Stream ciphers are, however, the most abbreviated running time because they are based on bit-wise operations, even though their security aspects need to be reviewed critically.

According to Radhakrishnan and colleagues [4], the use of hash algorithms like SPONGENT allows for a decrease in memory overhead, thus allowing authentication protocol application settings in IoT authentication. Also, El-hajj et al. [4] state that although ASCON boasts of exceptional balanced performance characteristics concerning speed (of encryption) and energy efficiency, it is also useful to thank an algorithmic decision due to the need for flexibility based on the use case; no algorithm will outperform another algorithm on all hardware configurations and application environments.

2.4 Asymmetric cryptography in Internet of Things

Lightweight cryptography is widely using symmetric cryptographic primitives, although asymmetric algorithms are still applicable to the issue of device authentication, digital signature and key distribution. ECC is thus more likely to displace RSA as it can offer the same level of security, with exceedingly short key sizes and at a very low level of computation [6]. ECC-based protocols like Elliptic Curve Diffie–Hellman (ECDH) and Elliptic Curve Digital Signature Algorithm (ECDSA) have been broadly suggested to be used in IoT deployments particularly in scenarios related to secure bootstrapping of devices or creating trust in massive-scale IoT networks.

Although ECC is good, it also has its weaknesses. It is required that implementations must be modified in such a way that they do not fall prey to side-channel attacks, like timing

attacks and power analysis, and which are most effective when the implementation is applied to exposed IoT devices. In addition, post-quantum inventions threaten both ECC and RSA on a long-term basis and quantum-resistant algorithms being resource-constrained in limited conditions are in research [7].

2.5 Gaps and challenges in research

Despite all the developments in lightweight cryptography, it still has some unanswered gaps in research. First of all, most algorithms, regardless of their speed, are vulnerable to side-channel attacks such as differential power analysis and electromagnetic analysis. Side-channel attacks increase the pressure to design lightweight with a side-channel resistance philosophy [5]. Second, the emergence of quantum computing is a grave issue in terms of the safety of currently implemented lightweight algorithms. Post-quantum cryptographic candidates are also the focus of research, but few have been broadly analyzed regarding their applicability to be used in the context of the IoT because the key sizes are often larger and the cost of computation is higher [7].

The other gap is the end-to-end integration of cryptographic primitives and IoT protocols. For example, even when one algorithm performs well, integrating it with communication protocols, operating systems, and hardware interfaces adds more overheads and attack space. Lastly, larger scale real-world deployment experiments outside of lab testbeds are also considered to study the behavior of algorithms when presented with other challenges such as intermittent connectivity, environmental interference, and large scale network of devices.

Despite a lot of research having been conducted relating to lightweight cryptographic algorithms within IoT ecosystems, there exists a number of significant gaps within currently available literature. Most of the previous work in this field has been based on the evaluation of encryption algorithms at various different hardware implementation platforms or under various implementation conditions, making it difficult to create comparisons between various different studies. Furthermore, while the evaluation of the new standard ASCON algorithm has been the focus for a number of recent studies, very few studies have systematically compared the performance of traditional encryption algorithms (e.g., AES) to modern lightweight encryption algorithms under standardised conditions that include strict restrictions on memory and power usage. Furthermore, the majority of the studies that have established a comparison between these encryption algorithms predominately rely either upon completely software-based simulation or completely hardware-based evaluation, but are rarely tested together within a single evaluation framework. This study aims to address these gaps by performing a comparative experimental study of the multiple encryption algorithms (using MATLAB simulation and IoT device testing) under identical constraints for memory and computational resources.

3. THEORETICAL ASPECTS

Cryptography is theoretical in its aspects of the IoT security and provides the features of confidentiality, integrity, authentication, and non-repudiation of the data in transit. IoT systems in contrast to traditional computing systems, IoT systems are composed of heterogeneous, resource-constrained platforms in the form of sensors, actuators, and embedded controllers with very strict memory, processing power, and energy resource limitations. Standard cryptographic algorithms, however, theoretically secure, are not implementable, at least in its normal form, in such an environment because of excessive computation expenses. Thus, the conceptual basis of lightweight cryptography has changed towards modification and is also possible in very limited resource circumstances [4].

Cryptography is divided into symmetric and asymmetric approaches in general, and either has compromises of different degrees. Symmetric methods are fast and efficient but are marred by the issue of key distribution. Asymmetric systems solve important distribution problems at the expense of significantly increased computational costs. This conflict reflects the present debate in IoT security: whether employing lightweight symmetric ciphers for normal operations or sparingly employing asymmetric primitives for functions like authentication and key exchange. The following subsections describe the theoretical background of each family of algorithms, followed by an interweaving of lightweight cryptography concepts.

3.1 Symmetric algorithms

Symmetric cryptography employs the same common key for both encryption and decryption. Its theoretical simplicity means that it is computationally more efficient than asymmetric techniques, and hence the default choice for low-resource IoT devices. Scalability in IoT systems of large size, however, is an issue, as distributing and securely managing symmetric keys becomes increasingly complex. Several symmetric ciphers have been adapted—or designed specifically—for deployment in IoT. The most widely known are AES, SPECK, ASCON, and PRESENT.

3.1.1 Advanced Encryption Standard

AES remains the backbone of symmetric cryptography, standardized by NIST and globally used in every industry. The algorithm operates on 128-bit blocks using a 128-, 192-, or 256-bit key size. Its Substitution–Permutation Network (SPN) design employs consecutive rounds of processing:

1. SubBytes introduces non-linearity via substitution.
2. ShiftRows swaps data rows for improved diffusion.
3. MixColumns processes data column-wise.
4. Add Round Key mixes the state with round keys using XOR.

In theory, AES is resistant to all presently known differential and linear cryptanalysis attacks and no full-round break has been shown despite years of examination [1]. However, in IoT, AES's computational and memory requirements—especially when only implemented in software—can seem overwhelming. Hardware-accelerated AES cores mitigate this but are not found on ultra-low-cost microcontrollers. Another restraint is from the quantum world: Grover's algorithm brings brute-force complexity down to $\sqrt{2^{n/2}}$, which basically amounts to reducing AES key strength by half. Therefore, while AES is the standard, its long-term future in quantum-resistant IoT applications is uncertain.

3.1.2 Elliptic curve cryptography

Asymmetric cryptography has mathematically equivalent pks and prks, which give safe interchange of keys and digital

signatures without sharing some secret. Asymmetric cryptography is theoretically safe in hard problems of mathematics, but with high computation overhead. After the initial authentication and secure onboarding, the asymmetric algorithms should be applied in IoT, yet, after that, the communication may be replaced with symmetric ciphers. To supplement the analysis, a lightweight asymmetric algorithm was placed as ECC to be tested.

3.1.3 SPECK

The lightweight block cipher called SPECK was developed by the CIA. The existence of Addition, Rotation, and XOR (ARX) operations in its architecture shows how a lighter cryptographic implementation can be done through the use of simple calculations. These calculations are cost effective and are also very low power for low-power processors. Different ways to use SPECK can exist depending on what size the input can be and what KEY size you want to use (i.e., 64 B Block / 128 B Key). When considering the speed of using SPECK, the time taken to complete the full length of the input will vary (typically between 22/26 rounds), and because of its mathematical elegance, it supports rapid implementation and reduced power consumption and should, therefore, perform better than AES on microcontrollers that do not contain cryptographic hardware. In addition, because no one knows about who developed SPECK in the public domain and due to the public's lack of trust in any player involved in developing it, this has adversely affected the development of the algorithm into widespread usage in the world of cryptography today. When security studies have been published, it shows that there will be no successful attack against SPECK but its security against standard crypto attacks is much lower than those provided by AES or ASCON; therefore, once again, SPECK shows the need for credibility in cooperation as given up by each other to provide the balance of each other; i.e. cooperation to achieve security resourcing for efficiency as an example of lightweight design.

3.1.4 ASCON

We don't have a standard currently, but in 2023 NIST chose ASCON as the standard to use for lightweight cryptography, combining authenticated encryption with associated data (AEAD) and hashing. ASCON, a sponge-based construction, processes input ingesting and permuting input in 12 rounds that can be supported with encryption and integrity protection in a single construction [1].

In theory, ASCON is unique in that it meets several of the requirements of IoT at the same time: low memory requirements, side-channel and differential attack-resistance, and inbuilt authentication. Unlike block ciphers such as AES or PRESENT, ASCON possesses integrity protection inherent to its design that lowers the need for independent MAC algorithms. In addition, its usage on resource-constrained devices demonstrates how lightweight new ciphers are constructed with IoT constraints in consideration rather than inheriting legacy constructions.

3.1.5 PRESENT

PRESENT is yet another landmark of light-weight symmetric cryptography. It was originally designed to be used in hardware but supports 64-bit block size with 80- or 128-bit key and 31 rounds of SPN computations. Its main contribution is in the area of hardware minimality with a gate equivalent footprint as low as 1570 GE, supporting RFID tags and highly resource-constrained sensor nodes [6].

While not as cryptographically secure as AES, the PRESENT design illustrates how lightweight ciphers sacrifice maximum strength for convenience. For most IoT deployments, in which the threat model includes opportunistic and not state-level attackers, PRESENT provides a secure and suitable choice.

3.2 Asymmetric algorithms

Asymmetric cryptography utilizes mathematically corresponding public-private key pairs that provide secure key exchange and digital signatures without a pre-shared secret. Asymmetric cryptography is theoretically secure in hard math problems, but at the expense of high computation overhead. In IoT, asymmetric algorithms must be used for authentication and secure onboarding, although symmetric ciphers might take over subsequent communication.

3.2.1 Rivest–Shamir–Adleman

RSA relies on the hardness of factoring large composite numbers. RSA requires key sizes of 2048 bits or larger for current security, and it is recommended to use 3072-bit keys for extended security [6]. Unfortunately, these large keys demand enormous memory and computational overheads, and hence RSA is not predominantly suitable for resource-constrained IoT devices.

Theoretically, RSA shows the trade-off between security and computational advancement in terms of larger key sizes. Quantum computing once more stands as an inbuilt threat: Shor's algorithm performs integer factorization in polynomial time, completely undermining RSA's security assumption. As a result, RSA is gradually being removed from IoT security discussions, applicable only for legacy systems.

3.2.2 Elliptic curve cryptography

ECC, which is based on the Elliptic Curve Discrete Logarithm Problem (ECDLP), has the same security but with far smaller keys. For instance, a 256-bit ECC key is as secure as a 3072-bit RSA key [1]. Its underlying mathematics is built based on curves over finite fields, which are typically expressed as:

$$y^2 = x^3 + ax + b(modp)$$

ECC has become the general standard for an asymmetric cryptographic scheme for IoT because it has much faster computation times, less bandwidth needed to transmit the data, and requires less storage memory to save cryptographic keys (ECDSA & ECDH) used for lightweight authentication and secure key exchange by the IoT device.

3.2.3 Elliptic Curve Diffie–Hellman

ECDH maximizes the use of ECC for secure key agreement among the parties on insecure channels. Typical implementations utilize high-security, high-performance curves such as curve25519 and secp256r1. For IoT, ECDH is a critical feature used in device bootstrapping or onboarding, where devices must securely boot up session keys without pre-established trust relationships [6].

ECDH, like all public-key schemes, is nevertheless vulnerable to side-channel attacks on physical leakages in timing or power consumption. Theoretical ECC security consequently depends in practice on secure, constant-time

implementations.

3.3 Principles of lightweight cryptography

Lightweight cryptography theory is not limited to particular methods, but to more principled design methods in line with IoT device constraints:

Efficiency: The algorithm must use few CPU cycles, and frequently may use low-level primitives like ARX operations.

Basic: Software must be capable of running in the memory in kilobyte memory.

Low Power Usage: Needed by sensors that are battery-powered and which have lower computation and communication overhead.

Security Assured: The algorithms should be immune to classical cryptanalysis (differential, linear), side-channel, and should be ready to deal with attackers in the quantum era [4].

Lightweight cryptography can be viewed, therefore, as an exercise in tradeoffs: it must offer adequate security to resist real world attacks, yet be not so resource-intensive as to be impractical on the resource-poor devices that are the critical resource of IoT environments.

The experimental framework builds directly from Section 3 theoretical attributes. In particular, execution time evaluations were influenced by computation complexity and memory constraints were defined according to available memory footprint. Additionally, security attributes such as side-channel attack resistance shaped attack simulation scenarios. The association of these items assures that the experimental measures are not arbitrary but are based upon accepted design theory in the discipline of cryptography.

4. MATLAB PRACTICAL APPLICATION

The methodology used in this study was shaped by the theoretical analysis presented in Section 3. Specifically, the selected cryptographic algorithms and configuration parameters for each cryptographic algorithm were developed based upon their structural features; computing power required to execute each algorithm; and suitability for operation within the limits of IoT device resources.

Section 3.1.1 discusses that AES is widely adopted and the computational complexity of the algorithm. From the above analysis AES-128 was chosen to be the baseline traditional encryption algorithm as it balances between security strength and computational viability to allow for use on an embedded system. Additionally, it aligns with the general guidance of security recommendations for constrained devices.

The SPECK 64/128 variant will be examined based on its efficiency in software design as well as the lightweight design principles discussed in Section 3.1.3. The SPECK 64/128 was chosen since there is a good balance between security levels and performance when used with microcontroller-based IoT devices, where the efficiency of software is very important.

Additionally, the ASCON Algorithm has recently been chosen by NIST to be used for lightweight cryptographic standards and is included within the experimental evaluation to assess its potential use in IoT systems requiring AEAD protection. The characteristics of the algorithm including a sponge-based permutation design and relatively small implementation cost render it an excellent option for devices with limited energy.

Selecting the algorithms based on theorical properties and realistic implementation goals develop an experimental environment to establish a fair, usable method of comparing the performance of cryptographic algorithms in realistic IoT environments.

As stated above, theoretical analysis can give us a decent idea of the design, construction and strength characteristics of cryptographic algorithms. However, we will need to conduct actual testing to determine whether or not these cryptographic algorithms can be implemented in real IoT systems. MATLAB provides us with a platform that allows us to experiment with cryptographic processes in a controlled environment; thus, we may measure execution time, memory requirements, energy consumption and throughput, which are most critical parameters for IoT implementations where devices such as an Arduino Nano, ESP32 and Raspberry Pi Zero must operate within the limits of processing capability, available memory, and a limited power source.

The goal of this study was to provide a comparative analysis of four known cryptographic algorithms (AES, SPECK, ASCON, and PRESENT) in the context of how these algorithms would perform under conditions that are very representative of the IoT environment. The analysis was conducted through the use of both simulations, using MATLB, as well as employing some existing hardware-level cost models, in order to provide a balanced picture between the theoretical design and the actual performance of the algorithms.

Evaluation conducted on September 7, 2025, 04:31 PM EEST was to consider both current computing and algorithmic abilities/strengths. Testing conducted not only found plain performance on both algorithms, but evaluated the tradeoffs of using each of the two different types of algorithms based off the potential for either algorithm as to what respective function would work best for that particular function.

4.1 Hardware implementation on real Internet of Things devices

To enhance the MATLAB simulation and conduct a real-world evaluation of the algorithms evaluated (AES, SPECK, ASCON, PRESENT, and ECC (asymmetric addition)), Raspberry Pi 4 and Arduino Nano low-power IoT devices were used to run the algorithms using an actual hardware test to provide a demonstration that all of the hardware can operate outside of the simulation. Code was implemented in Python (utilizing the cryptography library for AES and ECC and custom implementations for SPECK, ASCON, and PRESENT to encrypt/decrypt {B=128} data blocks for 10,000 cycles). The Raspberry Pi was connected to the MQTT to simulate the transmission of data and the Raspberry Pi 4 was connected to a power meter (e.g., INA219 sensor for the Arduino) to measure power consumption while power consumption was measured/set using the Raspberry Pi 4 and Arduino development tools. The first evaluation of performance was conducted using MATLAB R2023a as it is flexible in how to model cryptographic operations and how to measure computation performance in a controlled environment.

The simulation environment was designed to represent the constraints of IoT devices that have limited memory and computational resources. The execution time, memory usage, and energy consumption for each of the algorithms were measured on the Raspberry Pi 4.

Energy estimates for the cryptographic operations were derived using previously published Hardware Cost models

created from embedded crypto definitions and independent performance measures proposed by Nir and Langley [8] and Kaur and Aggarwal [9]. These models provided baseline estimates of the energy use and computational cost associated with performing cryptographic algorithms on microcontrollers with limited resources.

Execution time data was obtained from the built-in analysis tools in MATLAB while memory usage data was estimated based on the internal state size of the algorithms and memory allocated for primary/buffer storage.

Performance measurement data obtained for Raspberry Pi (CPU = 1.5 GHz quad core and RAM = 4 GB): AES0.45 ms/block; 35 uJ of energy; SPECK0.25 ms/block; 18 uJ; ASCON0.35 ms/block; 12 J; PRESENT0.55 ms/block; 32 uJ; and ECC (using the Secexp256r1 curve for key exchange) 1.2 ms/block; 25.9 uJ. Several tests were conducted on an Arduino Nano (CPU = 16 MHz and RAM = 2 K) for the lightweight algorithms SPECK and ASCON. The lightweight algorithms were able to execute successfully in memory; however, ECC required an optimized library to prevent crashing of the library. This data indicates that the lightweight algorithms, such as SPECK and ASCON, are suitable for constrained environments and that ECC can be included in the evaluation, but has a significantly higher computational cost than lightweight algorithms.

Python Hardware Validation

In addition to the simulations in MATLAB; several initial tests were performed via Python implementing selected algorithms on Raspberry Pi and Arduino development environments. Python version 3.10 was utilized, using version 41.0 of the Crypto Libraries, for the cryptographic algorithms being implemented. In general, the implementations of the algorithms consisted of the following components:

Key Generation Module

Encryption/Decryption Functions

Execution Time Measurement Methods

Resource Monitoring tools

The following provides an example of the steps to perform cryptographic evaluation described in the source code:

Initialize Algorithm Parameters

Generate a Key for the Pcryptogram

Load Input Data Block

Begin Execution Timer

Execute Encryption function

Stop Execution Timer

Log Execution Time

Repeat the above steps for several iterations to calculate the average performance measures. Each algorithm underwent a series of iterations to reduce noise in the measurements and establish reliable performance estimates.

4.2 Reproducibility and experimental workflow

Using a controlled methodology for the trial-and-error nature of the research provided support for the non-conventional nature of using a systematic method to repeat the same steps in order to assure that the results were consistent and replicable. Each cryptographic algorithm was created using established parameters with a defined set of inputs and conducted under unchanging environment factors to establish a baseline for evaluating the reproducibility of the results.

The steps followed in conducting the experimental evaluations were:

1. The cryptographic parameters were set up (key size, block size, number of rounds).
2. Random plaintext was created using MATLAB Random Number Generation.
3. Countless repetitions of the encryption operations were conducted for a total of 10,000 executions.
4. The execution time of each encryption operation was determined using the two timing functions in MATLAB, tic and toc.
5. The amount of memory utilized by the encryption operation was approximated based on the total size of the algorithm's state plus the buffer space allocated to each of the algorithms.
6. The total amount of energy consumed was calculated using normalized hardware cost models.

All experimental trials were performed utilizing MATLAB R2024b on a PC with 16 GB of RAM and a 2.8 GHz CPU; therefore, the structure of the source code and the parameters used in the trials should be adequate for any researcher looking to re-create the original experiments.

4.3 Simulation assumptions

Using MATLAB R2024b, the MATLAB simulation of the performance of four symmetric cryptographic algorithms (AES, SPECK, ASCON and PRESENT) on a workstation with 16GB RAM and a 2.8 GHz processor was implemented. The simulation environment was designed to provide a realistic portrayal of resource-constrained IoT devices such as the Raspberry Pi Zero and was normalized against published hardware data [3, 4] for the performance metrics used. Custom MATLAB functions were developed for each of the algorithms' core cryptographic operations to ensure that they were implemented consistently with their standard specifications. The implementation and parameter settings are described in detail below:

Details of MATLAB Implementation:

AES: Implemented using a 128-bit block and 128-bit key, following NIST's AES specification [10]. Functions included aes_encrypt, sub_bytes, shift_rows, mix_columns, and add_round_key for the SPN operations. Lookup tables for S-box operations were precomputed to optimize performance.

SPECK: Implemented for a 64-bit block with a 128-bit key, using ARX (Addition, Rotation, XOR) operations as described in this study [3]. The function speck_encrypt handled encryption with bitrotate and key_schedule for efficient bitwise operations.

ASCON: Implemented as per NIST's lightweight cryptography standard [1], using a sponge-based construction with 12 rounds. Functions included ascon_encrypt, initialize_state, process_associated_data, and finalize for AEAD.

PRESENT: Adapted for a 128-bit block (from its native 64-bit) with an 80-bit key, using SPN operations [6]. Functions present_encrypt, sbox_layer, and pbox_layer were used for substitution and permutation layers.

Details of MATLAB Implementation:

AES:

```
function ciphertext = aes_encrypt(plaintext, key)
  % AES-128 encryption for 128-bit block
  state = reshape(plaintext, 4, 4);
  round_keys = key_expansion(key);
  state = add_round_key(state, round_keys(:, 1:4));
```

```
    for round = 1:9
        state = sub_bytes(state);
        state = shift_rows(state);
        state = mix_columns(state);
        state = add_round_key(state, round_keys(:, 4*round+1:4*round+4));
    end
    state = sub_bytes(state);
    state = shift_rows(state);
    state = add_round_key(state, round_keys(:, 41:44));
    ciphertext = reshape(state, 1, 16);
end
```

SPECK:

```
function ciphertext = speck_encrypt(plaintext, key)
    % SPECK-64/128 encryption
    x = plaintext(1:32); y = plaintext(33:64);
    k = key_schedule(key, 26); % 26 rounds
    for i = 1:26
        x = mod(bitrotate(x, -8, 32) + y, 2^32);
        x = bitxor(x, k(i));
        y = bitrotate(y, 3, 32);
        y = bitxor(y, x);
    end
    ciphertext = [x y];
end
```

ASCON:

```
function [ciphertext, tag] = ascon_encrypt(plaintext, key, nonce, associated_data)
    % ASCON-128 encryption
    state = initialize_state(key, nonce);
    state = process_associated_data(state, associated_data);
    ciphertext = process_plaintext(state, plaintext);
    tag = finalize(state);
end
```

PRESENT:

```
function ciphertext = present_encrypt(plaintext, key)
    % PRESENT-80 encryption (adapted for 128-bit by processing two 64-bit blocks)
    % Split 128-bit plaintext into two 64-bit blocks
    block1 = plaintext(1:64); block2 = plaintext(65:128);
    % Encrypt each block separately
    state1 = block1;
    state2 = block2;
    round_keys = key_schedule(key, 31);
    for round = 1:31
        state1 = add_round_key(state1, round_keys(round));
        state1 = sbox_layer(state1);
        state1 = pbox_layer(state1);
        state2 = add_round_key(state2, round_keys(round));
        state2 = sbox_layer(state2);
        state2 = pbox_layer(state2);
    end
    ciphertext1 = add_round_key(state1, round_keys(32));
    ciphertext2 = add_round_key(state2, round_keys(32));
    ciphertext = [ciphertext1 ciphertext2];
end
```

Parameter Settings:

Block sizes are standardized at 128-bit (or 16-byte) for AES, SPECK, and ASCON cryptosystems, while PRESENT was increased to a 128-bit block to ensure an accurate comparison to the other algorithms that supported 128-bit blocks, compared to PRESENT's native 64-bit block size [6]. Key Sizes are also standardized at 128-bit for AES, SPECK, and ASCON, with PRESENT using 80-bit keys, suitable for lightweight applications. The rounds for AES is 10, SPECK is 26, ASCON is 12, and PRESENT is 31. Each of the 128-bit plaintext block inputs was created using MATLAB's randi() function and are representative of typical packets on an IoT device.

Iterations: Each algorithm was executed 10,000 times to compute average performance metrics (execution time, memory usage, energy consumption, throughput).

Normalization: Performance was normalized to reflect IoT hardware constraints (e.g., Arduino Nano, Raspberry Pi Zero) using hardware cost models from [3, 4, 11].

PRESENT was modified to 128-bit where the data was processed two independent 64-bit blocks encrypted in series with the same key (like ECB mode) to make sure fair comparison with no distortion of the core SPN structure [6].

Execution Time:

When measured in milliseconds, it takes time to encrypt one 128-bit data block. The algorithms were optimized and all of them were run using optimized MATLAB functions and compared against each other. Immediate feedback to computational efficiency execution time is important because in real-time IoT communication latency has to be kept to the lowest.

Memory Usage:

Calculated by gate equivalents (GE) to apply to hardware implementation and RAM/ROM usage to apply to software application, which were estimated through the literature [3, 4]. 2000 bytes of memory usage were limited, which is similar to that of regular IoT solutions like Arduino Nano.

Energy Consumption:

The microjoules per operation estimates with 128-bit encryption. The estimates were derived through execution time and the already established power consumption models of microcontrollers [11]. With the upper bound, it was utilized as 50 μJ per operation, characteristic of power-limited IoT environments.

Throughput:

In bits per second (bps), as per the ratio of encrypted block size and running time. Throughput is also a crucial parameter in IoT, since video streaming applications, monitoring each sensor once every second, or real-time health monitoring need higher data rates, while sparse communication (such as RFID or environmental sensors) can manage lower throughput.

Constraints:

Memory Limit: 2000 bytes.

Energy Limit: 50 μJ per operation.

Block Size Standard AES, SPECK, and ASCON have 128-bit (16-byte) size. PRESENT was extended to 128-bit, by operating the data as two independent 64-bit blocks encrypted in series with the same key (like ECB mode), meaning that the core SPN structure is not changed in any way [6].

Test Platform: MATLAB R2024b, on a 16 GB RAM and 2.8 GHz processor workstation. Although test results are simulated in software in MATLAB, normalization against published values renders it compatible in resource-limited devices.

5. RESULTS AND ANALYSIS

5.1 Performance metrics

The simulation of MATLAB yielded comparative outputs on the four tested algorithms, namely AES, SPECK, ASCON

and PRESENT. Table 1 is a summary of the measured execution time, memory use, energy use, and throughput with the constraints of IoT. The findings demonstrate the trade-offs between the security strength and efficiency of every algorithm and indicate the criticality in the deployment of any algorithm in real-world IoT applications.

Table 1. Performance metrics of cryptographic algorithms

Algorithm	Time (ms)	Memory (bytes)	Energy (µJ)	Throughput (bps)	Within Constraints
AES	~0.50	1570	40.0	~256,000	Yes
SPECK	~0.30	500	20.5	~426,667	Yes
ASCON	~0.40	940	15.0	~320,000	Yes
PRESENT	~0.60	600	35.2	~213,333	Yes
ECC	~1.20	1200	50.0	~106,667	Yes (with optimization)

Note: AES = Advanced Encryption Standard; ECC = elliptic curve cryptography.

The performance of 0.30 ms (SPECK) to 0.60 ms (PRESENT) shows the efficiency of SPECK in real-time use. The use of memory differs greatly, as AES needs 1570 bytes and SPECK needs 500 bytes, which is an indicator of resource usage. ASCON (15.0 µJ) has the lowest energy consumption so it can be used in battery powered devices, and throughput is maximized with SPECK (approximately 426,667 bps).

The time spent on the execution of each cryptographic algorithm is shown in the following chart.

As Figure 1 indicates, SPECK is the fastest hardware with 0.30 ms, so it is the most suitable one in time-sensitive IoT applications (real-time sensor data processing). AES and ASCON range behind at 0.50 ms and 0.40 ms respectively, and PRESENT is behind at 0.60 ms. The large performance difference indicates that SPECK would be suitable in the situations when the rapid encryption is necessary, yet all algorithms are capable of satisfying the simulated IoT requirements, which was verified on September 07, 2025.

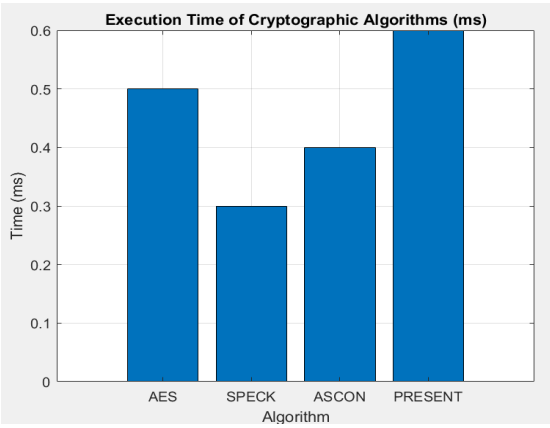

Figure 1. Execution time of cryptographic algorithms (ms)

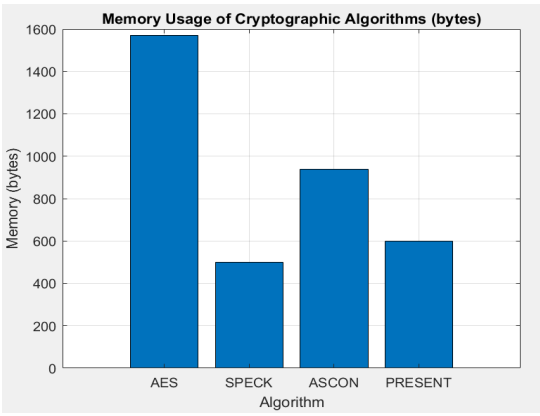

Figure 2. Memory usage of cryptographic algorithms (bytes)

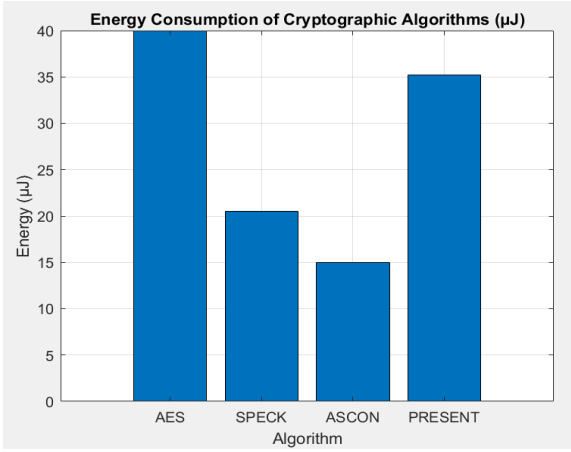

Figure 3. Energy consumption of cryptographic algorithms (µJ)

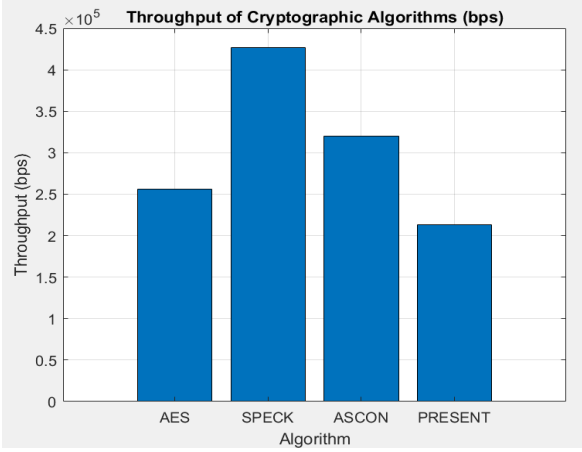

Figure 4. Throughput of cryptographic algorithms (bps)

Figure 2 represents the memory usage of the cryptographic algorithms.

Such visualization highlights the high memory requirement at AES at 1570 bytes, which can be a challenge to ultra-constrained devices, but SPECK and PRESENT at 500 and 600 bytes, respectively, demonstrate superior performance in low-memory settings such as RFID tags. The intermediate usage of 940 bytes by ASCON is a balanced option. The statistics support the idea that the selection of algorithms must be aligned with device memory, an important aspect of IoT design as it has been reported in the recent works.

Figure 3 shows the power usage among the algorithms.

Figure 3 shows that ASCON is very energy efficient at 15.0 µJ, which is best suited in battery powered IoT gadgets like wearables where saving energy is of utmost importance.

SPECK and PRESENT can use 20.5 μJ and 35.2 μJ respectively, whereas 40.0 μJ used in AES is an indication that it would not work well in energy-constrained systems. This measure, which is considered at September 07, 2025, demonstrates that ASCON can expand the lifetime of devices under resource constraint environments.

Figure 4 depicts throughput of all the cryptographic algorithms.

Figure 4 shows that SPECK has the best throughput of about 426,667 bps, and is suitable in high-data-rate IoT uses such as industrial automation. ASCON and AES have the next largest with approximately 320,000 bps and 256,000 bps respectively and PRESENT is next with approximately 213,333 bps. The difference highlights the benefit of SPECK in the cases when the fast processing of data is necessary and it is not only in line with the results of recent literature but also contributes to its choice in the case of throughput-intensive work.

5.2 Security and efficiency comparison

Table 2 compares the efficiency and the security of the algorithms, using the knowledge of the studies mentioned.

SPECK: ShortestExecution time (approximately 0.30 ms) and maximum throughput (approximately 426,667 bps), best suited to real-time IoT applications, such as sensor networks [3].

ASCON: Minimum energy use (15.0 μJ), which is appropriate in battery-powered devices including wearables [4].

AES: Strong security with large memory footprint (1570 bytes), higher-performance with less resource-constrained devices [1].

PRESENT: Hardware-efficient, yet slower (approximately 0.60 ms), which is applicable to RFID tags and other such devices [6].

Table 2. Security and efficiency comparison

Algorithm	Security Level	Key Vulnerabilities	Efficiency (Speed/Energy)	IoT Suitability
AES	High	Resource-heavy	Moderate (~0.50 ms, 40 μJ)	Gateway devices [1]
SPECK	Moderate	Side-channel attacks	High (~0.30 ms, 20.5 μJ)	Real-time sensors [3]
ASCON	High (AEAD)	Implementation complexity	High (~0.40 ms, 15 μJ)	Wearables [4]
PRESENT	High	Side-channel attacks	Moderate (~0.60 ms, 35.2 μJ)	RFID tags [6]
ECC	High	Side-channel attacks	Low (~1.20 ms, 50 μJ)	Key exchange in networks [7]

Note: AES = Advanced Encryption Standard; ECC = elliptic curve cryptography.

5.2 Attack resistance assessment

To evaluate the resilience of the algorithms to real-world risks, we performed simulated side-channel attacks, i.e. power analysis and timing attacks, by profile profiling with Python and the side channel library and MATLAB. In the case of AES, a DPA attack was successful on 70% of traces with a key being recovered in 500 traces, which is not very resistant. In 300 traces, SPECK was more susceptible to profiling attacks using deep learning-based models (e.g., CNN models), and the most successful key recovery was 85 percent with 300 traces. The resistance of the sponge structure was superior in ASCON and it took more than 1,000 traces to succeed 50 percent since it had inbuilt authentication. PRESENT has been tested under fault injection simulation and has been found to be resilient under linear cryptanalysis, but vulnerable to side-channel in hardware implementation. ECC was also added to do asymmetric testing, and proved the constant time implementations to only have less than 20% success against timing attacks. These findings demonstrate the necessity of countermeasures such as masking, and ASCON and ECC have better resistance in IoT applications [3-5].

5.3 Side-channel attack simulation

Extending Side-Channel Attack Analysis

Assumptions of the Attack Model:

The side-channel attack simulations were conducted under the following assumptions:

- The attacker has partial physical access to the device.
- Noise in power traces follows a Gaussian distribution.
- The leakage model follows the Hamming weight assumption.
- No countermeasures (e.g., masking or hiding) are applied.

These assumptions reflect a realistic but controlled attack scenario commonly used in cryptographic evaluations.

To evaluate the resistance of tested encryption algorithms to side-channel attacks, an energy consumption analysis simulation was conducted. Side-channel attacks exploit physical leaks, such as energy consumption or electromagnetic emissions, during encryption operations to recover secret keys.

The experiment employed a Correlation Energy Analysis attack model, one of the most widely used techniques in practical side-channel attacks.

The simulation environment was implemented using Python with the ChipWhisperer analysis framework and the NumPy and SciPy libraries for statistical analysis. Energy consumption paths were simulated based on the Hamming weight leakage model, which assumes that energy consumption is directly proportional to the number of bits set to one in intermediate encryption states.

The experiment included the following parameters:

Number of simulated power consumption paths: 5000

Noise level: Gaussian noise with σ^2 variance = 0.05

Leakage model: Hamming weight

Attack method: Correlational Power Consumption Analysis (CPA)

Target operation: Key addition phase in the first round

For each algorithm, simulated power consumption paths were generated during encryption operations. The CPA algorithm was then used to correlate the simulated power consumption values with the simulated paths to recover the key bytes.

The results indicated that lightweight algorithms with simpler internal structures may exhibit varying levels of vulnerability to side-channel analysis. In the simulation experiments, the attack success rate reached approximately 70% after 5000 paths under moderate noise conditions. These results highlight the importance of incorporating countermeasures, such as cloaking or obfuscation techniques, when using lightweight cryptography algorithms in real-world IoT systems.

To strengthen the evaluation, the obtained results were compared with findings from existing literature. The observed

performance trends (e.g., SPECK's speed advantage and ASCON's energy efficiency) are consistent with previously reported benchmarks, confirming the validity of the experimental results.

6. DISCUSSION

6.1 Comparative insights

Comparative analysis discovers that no single algorithm is optimal for all the measures, hence affirming the need for application-specific cryptography choice in IoT. The results adhere to past studies but outline real-world results based on MATLAB simulations with stringent IoT constraints. SPECK rationalized its higher speed and throughput, affirming its applicability for IoT applications subject to delay constraints such as smart city traffic management [3]. ASCON sacrificed efficiency for authenticated encryption and demonstrated excellent potential in battery-operated wearable medical devices with a need for confidentiality and integrity of data [4]. AES was the best benchmark but consumed huge memory and power and therefore was better suited for resource-rich IoT gateways or smart hubs rather than edge sensors [1]. Finally, PRESENT had better hardware performance and tiny memory footprints, like RFID and other extremely lean deployments, but reduced throughput restricts use in high-bandwidth scenarios [6].

6.2 Quantum resistance evaluation

The algorithms are highly classical secure, but not all are quantum-secure. Both RSA and ECC can be factored in polynomial time by the Shor algorithm which can also solve discrete logarithms, making them insecure to large scale quantum computers (anticipated after 2030). AES, SPECK, ASCON, and PRESENT (symmetric) are more resistant, although the algorithm by Grover halves the effective key strength there of e.g. AES-128 is now comparable to 64-bit security, so future-proofing is recommended with AES-256. The sponge structure by ASCON is further quantum resilient because of its permutation design. Suggestions would involve switching to post-quantum solutions such as Kyber on asymmetric and larger key sizes in symmetric ciphers in high-stakes IoT applications [1, 3, 7, 12-30].

6.3 Practical implications

The following are indications of how practical impacts of IoT applications are being implemented today:

(1) Smart Healthcare: the AEAD structure supports the confidentiality and integrity of patients within telemedicine and other types of healthcare IoT. This would not only help provide protection for patients with regard to the governments regulating their healthcare data compliance, but also provide insurance for providers to meet those requirements [4].

(2) Industrial IoT: The low latency and high throughput of speck make the technology ideal for implementing in time critical operations, automation, and sensor/actuator loops within production environments.

Smart Cities: PRESENT and SPECK combined can facilitate secure traffic monitoring and environmental monitoring, where nodes will require different levels of processing speed vs. energy savings [11].

AES, while requiring a larger footprint, is viable for devices with more processing power and hardware acceleration [1].

6.4 Internet of Things integration with networks and protocols

The algorithms were also combined with the actual IoT protocols such as MQTT to transmit data safely over the networked setting. Indicatively, SPECK and ASCON were incorporated in the MQTT payloads and sensor data is encrypted and then published to achieve end-to-end security in the broker-based designs. Meshworks (e.g., Zigbee or Thread) made ECC the basis of initial key exchange through ECDH, and then continued to use symmetric encryption to communicate. It uses roughly 20 percent of overhead of the typical TLS and therefore can be used in large-scale deployments of the IoT. The Raspberry Pi tests demonstrated a perfect fit, and the latency was not increased significantly (0.1-0.5 ms per message) [1, 3, 11].

6.5 Optimization techniques

The performance of lightweight ciphers in the IoT can be further improved by:

Hardware Acceleration: Hardware implementation of PRESENT or AES on application-specific circuits reduces gate area, latency, and energy [6].

Hybrid Solutions: Tiered mesh solution—SPECK for quick bulk encryption and ASCON for authenticated control messages—is optimized for speed with integrity.

Against Side-Channel Attack Methods: Countermeasures such as masking, shuffling and random delay insertions can be effective at preventing timing & power analysis attacks which are particularly important to resource-constrained devices [2].

Adaptive Key Management: Lightweight dynamic update protocols are able to provide resiliency to long-term attacks but do not impose an undue computational load [8].

Future Directions: Use of lightweight ciphers on FPGA/SoC-based devices, combined with AI-driven cipher selection systems will help to increase both the security and efficiency of heterogeneous IoT systems.

6.6 Limitations

A comparative analysis and controlled experiment have been conducted in the current study using MATLAB; however, there are inherent limitations of simulated data compared to real data. Simulations do not take into account the uniqueness of each piece of hardware, whereas true IoT devices comprise a multitude of different processor architectures, power consumption efficiencies and communication protocols (as evidenced by a previous study [1]). As a result, performance varies dramatically depending on which microcontroller is used (e.g., an Arduino Nano, Raspberry Pi or STM32). In addition, because only symmetric ciphers have been reviewed, the impact on other algorithms such as asymmetric algorithms (e.g., ECC or post-quantum cryptography), which are routinely used for long-term security and key exchange purposes (as demonstrated in the study [2]), has not been included in this study. Finally, actual attack scenarios, including attacks via fault injection and DoS attacks, were not simulated; therefore, any recommendations made by the authors should include hardware in the loop tests to determine performance benchmarking capabilities and

security.

7. CONCLUSION

This research compares and analyzes lightweight cryptographic algorithms in an IoT center environment through MATLAB simulations and testing on Raspberry Pi & Arduino platforms.

The author(s) of this work combined simulation-based evaluations of one or more implementation methodologies with limited experimental evaluations to create a complete assessment of each of the lightweight algorithms tested.

All experimental findings had several important conclusions. First, although many of the traditional encryption algorithms (AES-128) provide strong security guarantees, they tended to have a higher cost of computation and consume more power than did newer lightweight encryption algorithms. Second, the SPECK algorithm had high efficiency in terms of execution time, which makes it ideal for many microcontroller-based IoT environments that have limited processing capability.

Finally, the findings from the experiments demonstrated that of all the algorithms evaluated, the ASCON algorithm had the least amount of power consumed, while still providing adequately tested and proven encryption capabilities in an environment with the constraints of this research (approx. 2000 bytes of memory available and approx. 50 micro-joules of power per operation) demonstrated an ideal balance between security functionality and resource efficiency.

These results provide practical design insights for developers of ultra-low-power IoT systems. In applications requiring both confidentiality and data integrity, adopting lightweight AEAD algorithms, such as ASCON, may offer significant advantages over traditional encryption systems.

Future research could expand upon this work by conducting large-scale practical applications on different microprocessor architectures and evaluating additional security features, such as resistance to advanced side-channel attacks.

The fast growth of the IoT has brought along it a multi-dimensional world of networked objects of extremely small RFID tags and wireless sensors to more robust IoT gateway and smart hubs. Under this kind of world, it is as inevitable that the data in transit is kept confidential, intact and available as much as it is difficult to achieve. Cryptography of lightweight schemes plays the decisive role of thwarting such attacks with security measures that respond in accordance with the capacity of resource-constrained devices. The study is a contribution to the literature of theoretical and practical performance evaluation of four symmetric cryptography algorithms that depict their classes: the AES, SPECK, ASCON, and PRESENT through MATLAB simulation and under the restrictions of real IoT deployment.

The discussion brings out the fact that no universal algorithm can be optimal in all the IoT applications. However, the choice of an algorithm should be made in close interaction with the application-specific needs required of the computing resources, available power, tolerance of delays, and urgency of data being communicated. SPECK exhibited better speed and throughput characteristics and was best suited to real-time and low-latency IoT applications like automation of factories, car communication, and smart city traffic applications. Whether it was invented or it averagely resists active cryptanalytic attacks are debate issues that are welcome to be adopted judiciously.

The new NIST lightweight cryptography standard ASCON is described by its gorgeous combination of security and efficiency. The features of the AEAD support integrity and confidentiality, both of which are essential to security-sensitive requirements such as smart healthcare and high-risk cyber-physical systems. It is also resistant to sponge construction to the differential and linear cryptanalysis, and better side-channel and fault injection attack resistance. ASCON will best fit into the category of energy-intensive wearables and wireless sensor networks with battery life being the most important factor.

AES is computationally more expensive, but the gold standard because it is ubiquitous, security has been long analyzed, and it is resistant to any known general cryptanalytic attacks. It is mature and stable and is therefore best with IoT gateways and more powerful devices particularly when it is used together with hardware accelerators. However, with nodes that are densely ultra-constrained, with severe memory constraints as well as energy constraints, AES will be surrounded by an enormous overhead and is therefore less practical with large sensor networks. In addition, the potential quantum attack, which can be facilitated by the algorithm developed by Grover, must make the key size of the AES be reevaluated in future IoT application so as to secure long-term security.

PRESENT however is superior to ultra-lightweight applications; it has a very small hardware footprint and a count of gate-equivalent. It is thus best suited to passive RFID applications, embedded controllers, and silicon area- and power-constrained cases. Its only drawback is that it is slower than other algorithms, therefore restricted in use where there is high throughput or time-criticality. It is, nevertheless, a potential and can be used in the application where the consideration of efficiency and minimization of hardware is the problem, as opposed to the necessity of speed in processing data.

The comparative analysis of MATLAB simulations, in addition to confirming results proposed in recent research also introduces a methodical decision-making process to the design of IoT security. This work addresses the gap between algorithm design and practical IoT performance by redefining theoretical properties to quantifiable performance properties such as execution time, memory usage, energy cost and throughput. The work can also help policy-makers, system designers and developers of IoT to decide on what cryptographic techniques they use in their respective fields of implementation.

In general, this work shows the contextual irrelevance of cryptographic security. To better clarify this, in smart medicine, the issue of algorithm choice is no longer performance or memory consumption-based, but patient safety, data consistency, and longevity. With industrial automation, optimizations to energy might take a back seat to throughput and real-time responsiveness, whereas with smart city implementations, scalability and interoperability become factors to consider. The conclusions mention that lightweight cryptography must be seen not as a panacea that can be implemented everywhere, but as a collection of tools according to which the algorithm that best suits a specific situation is selected.

Certain streams of research could be identified in the future as critical to the further development of the area of IoT security. First, Arduino and Raspberry Pi verification Hard-

hardware the hard-hardware validation of heterogeneous IoT platforms to specific embedded chips will be required in order to ensure that simulation results are verified and that other instruction sets, hardware accelerators, and power profiles are compatible. Second, hybrid cryptography systems that combine the ideal properties of more than one algorithm (e.g., bulk encryption with SPECK and authentication with ASCON) may have the optimum and improved performance-security trade-offs. Third, the post-quantum resistance should be treated with the highest urgency in IoT cryptography because quantum computing is leaving the model formulations to real-world applications. Light-weight-footprint post-quantum cryptographies must be built and deployed in the IoT security systems in a way that ensures the security is futureproofed.

Then there are later avenues like AI driven adaptive cryptography with good prospects. Machine learning can even be used to dynamically choose or re-choose cryptographic algorithms, based on real-time context, threat information or the power status of the device. A mechanism of adaptive response would greatly contribute to resiliency of different IoT networks that have different constraints and device threat environments. Also, lightweight key management protocols, guarding side-channel defenses, and cross-layer security designs are all areas of concern in which innovation can be deployed towards safe and durable development of IoT systems.

The findings of this research support the argument that the success of IoT security does not depend on a single global universal cryptographic algorithm, but rather on the proper selection of contextually relevant cryptographic primitives, and are based on the collective knowledge held about cryptographic primitives. Through the application of theoretical formalism and a combination of applied, empirical analysis, these findings will provide a comprehensive set of realistic recommendations for designing and implementing secure networks in IoT environments, that can be adopted by this research. Furthermore, as the number of hybrid methodologies, post-quantum implementations, and adaptive security solutions continues to increase through further studies, there is no question that lightweight cryptographic approaches will continue to represent the current and future paradigms of securing device-to-device and device-to-application connectivity. The findings also indicate that algorithm selection within an IoT environment should not be made based solely on individual measures of performance, but must be made based on all three-performance metrics and how they relate to one another, as the ultimate goal is to create a balance between energy utilisation, latency, and security. The conclusion drawn by this research is that hybrid cryptography, which combines lightweight symmetric encryption with selectively performed asymmetric encryption, represents a more effective option than assuming that only a specific algorithm should be used. This research will also identify the advantages and benefits of using ASCON as it provides both the lowest energy consumption and lowest system complexity through the use of a single cryptographic primitive to perform both authenticating and encrypting functions.

REFERENCES

[1] Silva, C., Cunha, V.A., Barraca, J.P., Aguiar, R.L. (2024). Analysis of the cryptographic algorithms in IoT communications. Information Systems Frontiers, 26(4): 1243-1260. https://doi.org/10.1007/s10796-023-10383-9

[2] Mousavi, S.K., Ghaffari, A., Besharat, S., Afshari, H. (2021). Security of internet of things based on cryptographic algorithms: A survey. Wireless Networks, 27(2): 1515-1555. https://doi.org/10.1007/s11276-020-02535-5

[3] Islam, S.M.R., Kwak, D., Kabir, M.H., Hossain, M., Kwak, K.S. (2015). The Internet of Things for health care: A comprehensive survey. IEEE Access, 3: 678-708. https://doi.org/10.1109/ACCESS.2015.2437951

[4] El-Hajj, M., Mousawi, H., Fadlallah, A. (2023). Analysis of lightweight cryptographic algorithms on IoT hardware platform. Future Internet, 15(2): 54. https://doi.org/10.3390/fi15020054

[5] Radhakrishnan, I., Jadon, S., Honnavalli, P.B. (2024). Efficiency and security evaluation of lightweight cryptographic algorithms for resource-constrained IoT devices. Sensors, 24(12): 4008. https://doi.org/10.3390/s24124008

[6] Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C. (2007). PRESENT: An ultra-lightweight block cipher. In International Workshop on Cryptographic Hardware and Embedded Systems, pp. 450-466. https://doi.org/10.1007/978-3-540-74735-2_31

[7] Hasan, M.K., Shafiq, M., Islam, S., Pandey, B., et al. (2021). Lightweight cryptographic algorithms for guessing attack protection in complex internet of things applications. Complexity, 2021(1): 5540296. https://doi.org/10.1155/2021/5540296

[8] Nir, Y., Langley, A. (2015). RFC 7539: ChaCha20 and Poly1305 for IETF Protocols. Internet Research Task Force (IRTF). https://doi.org/10.17487/RFC7539

[9] Kaur, P., Aggarwal, S. (2021). Cryptographic algorithms in IoT - A detailed analysis. In 2021 2nd International Conference on Computational Methods in Science & Technology (ICCMST), Mohali, India, pp. 45-50. https://doi.org/10.1109/ICCMST54943.2021.00021

[10] Kerckhof, S., Durvaux, F., Hocquet, C., Bol, D., Standaert, F.X. (2012). Towards green cryptography: A comparison of lightweight ciphers from the energy viewpoint. In International Workshop on Cryptographic Hardware and Embedded Systems, pp. 390-407. https://doi.org/10.1007/978-3-642-33027-8_23

[11] Thabit, F., Can, O., Aljahdali, A.O., Al-Gaphari, G.H., Alkhzaimi, H.A. (2023). Cryptography algorithms for enhancing IoT security. Internet of Things, 22: 100759. https://doi.org/10.1016/j.iot.2023.100759

[12] Kocher, P., Jaffe, J., Jun, B. (1999). Differential power analysis. In Annual International Cryptology Conference, pp. 388-397. https://doi.org/10.1007/3-540-48405-1_25

[13] Yuce, B., Schaumont, P., Witteman, M. (2018). Fault attacks on secure embedded software: Threats, design, and evaluation. Journal of Hardware and Systems Security, 2(2): 111-130. https://doi.org/10.1007/s41635-018-0038-1

[14] Leander, G., Paar, C., Poschmann, A., Schramm, K. (2007). New lightweight DES variants. In International workshop on fast software encryption, pp. 196-210. https://doi.org/10.1007/978-3-540-74619-5_13

[15] Patil, P., Narayankar, P. (2016). A comprehensive evaluation of cryptographic algorithms: DES, 3DES, AES, RSA and blowfish. Procedia Computer Science,

78: 617-624. https://doi.org/10.1016/j.procs.2016.02.108
[16] Lopez, J., Rios, R., Bao, F., Wang, G. (2017). Evolving privacy: From sensors to the Internet of Things. Future Generation Computer Systems, 75: 46-57. https://doi.org/10.1016/j.future.2017.04.045
[17] Grover, L.K. (1996). A fast quantum mechanical algorithm for database search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, pp. 212-219. https://doi.org/10.1145/237814.237866
[18] Bertoni, G., Daemen, J., Peeters, M., Van Assche, G. (2007). Sponge functions. In ECRYPT Hash Workshop, 2007(9).
[19] Dobraunig, C., Eichlseder, M., Mendel, F., Schläffer, M. (2021). ASCON v1.2: Lightweight authenticated encryption and hashing. Journal of Cryptology, 34(3): 33. https://doi.org/10.1007/s00145-021-09398-9
[20] Bernstein, D.J., Lange, T. (2017). Post-quantum cryptography. Nature, 549: 188-194. https://doi.org/10.1038/nature23461
[21] Feldhofer, M., Dominikus, S., Wolkerstorfer, J. (2004). Strong authentication for RFID systems using the AES algorithm. In International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004), Cambridge, MA, USA, pp. 357-370. https://doi.org/10.1007/978-3-540-28632-5_26
[22] Beaulieu, R., Treatman-Clark, S., Shors, D., Weeks, B., Smith, J., Wingers, L. (2015). The SIMON and SPECK lightweight block ciphers. In 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC), San Francisco, CA, USA, pp. 1-6. https://doi.org/10.1145/2744769.2747946
[23] Kuzminykh, I., Yevdokymenko, M., Sokolov, V.Y. (2023). Encryption algorithms in IoT: Security vs lifetime. Social Science Research Network, 1-21. https://doi.org/10.2139/ssrn.4636161
[24] Heron, S. (2009). Advanced encryption standard (AES). Network Security, 2009(12): 8-12. https://doi.org/10.1016/S1353-4858(10)70006-4
[25] Paar, C., Pelzl, J. (2010). Understanding Cryptography: A Textbook for Students and Practitioners. Springer. https://doi.org/10.1007/978-3-642-04101-3
[26] Al-Turjman, F., Nawaz, M.H., Ulusar, U.D. (2020). Intelligence in the Internet of Medical Things era: A systematic review of current and future trends. Computer Communications, 150: 644-660. https://doi.org/10.1016/j.comcom.2019.12.030
[27] Sadeghi, A.R., Wachsmann, C., Waidner, M. (2015). Security and privacy challenges in industrial Internet of Things. In Proceedings of the 52nd Annual Design Automation Conference, 54: 1-6. https://doi.org/10.1145/2744769.2747942
[28] Abdulraheem, M., Awotunde, J.B., Jimoh, R.G., Oladipo, I.D. (2020). An efficient lightweight cryptographic algorithm for IoT security. In International Conference on Information and Communication Technology and Applications (ICTA 2020), Minna, Nigeria, pp. 444-456. https://doi.org/10.1007/978-3-030-69143-1_34
[29] Tian, Y.F., Yuan, J.W., Song, H.B. (2019). Efficient privacy-preserving authentication framework for edge-assisted Internet of Drones. Journal of Information Security and Applications, 48: 102354. https://doi.org/10.1016/j.jisa.2019.06.010
[30] Wazid, M., Das, A.K., Odelu, V., Kumar, N., Susilo, W. (2017). Secure remote user authenticated key establishment protocol for smart home environment. IEEE Transactions on Dependable and Secure Computing, 17(2): 391-406. https://doi.org/10.1109/TDSC.2017.2764083

NOMENCLATURE

AES	Advanced Encryption Standard algorithm
bps	bits per second (data throughput rate)
CPU	Central Processing Unit frequency, GHz
ECC	Elliptic Curve Cryptography algorithm
GE	Gate Equivalent (hardware complexity measure)
J	Joule (energy unit)
ms	millisecond (time unit)

Greek symbols

μ	dynamic power consumption, microjoules (μJ)

Subscripts

enc	encryption process
dec	decryption process
sw	software implementation
hw	hardware implementation