Journal of Internet Services and Applications, 2026, 17:1, doi: 10.5753/jisa.2026.6502

PRISEC III: Dynamic Cryptographic Adaptation for Balancing Performance and Security

Humza Sohail [**Instituto Universitário de Lisboa (ISCTE-IUL), ISTAR, Lisboa, Portugal** | ***humza_sohail@iscte-iul.pt***]
Darlan Noetzold [**Expert Systems and Applications Laboratory, University of Salamanca, Salamanca, Spain** | ***darlannoetzold@usal.es***]
Valderi Reis Quietinho Leithardt [**Instituto Universitário de Lisboa (ISCTE-IUL), ISTAR, Lisboa, Portugal** | ***valderi.leithardt@iscte-iul.pt***]

Instituto Universitário de Lisboa (ISCTE-IUL), ISTAR, Lisboa, Portugal.

Received: 04 September 2025 • **Accepted:** 06 April 2026 • **Published:** 23 May 2026

Abstract
This paper introduces PRISEC III, a dynamic cryptographic framework designed to balance security and performance in heterogeneous IoT environments. Unlike static, one-size-fits-all approaches, PRISEC III employs a role-based multi-level model that adapts cryptographic strategies according to data sensitivity, device constraints, and network conditions. The framework integrates lightweight symmetric primitives for efficiency, robust asymmetric methods for secure key exchange, and hybrid schemes that combine multiple layers of protection. To enhance long-term resilience, PRISEC III also incorporates post-quantum options, ensuring preparedness against emerging cryptographic threats. Evaluation results demonstrate that the approach achieves scalable security while maintaining feasible computational costs, making it suitable for large-scale and resource-constrained IoT deployments.

Keywords: Adaptive cryptography, IoT security, lightweight encryption, authenticated encryption, key management, post-quantum cryptography

1 Introduction

The rapid expansion of the Internet of Things (IoT) has introduced significant security challenges due to the limited computational power of many IoT devices. These constraints often prevent the adoption of computationally intensive cryptographic mechanisms, leaving IoT applications vulnerable to various cyber threats, including unauthorized data access, data breaches, and denial-of-service (DoS) attacks Silva *et al*. [2024]; Yuan *et al*. [2024]. Ensuring the confidentiality, integrity, and authenticity of data in IoT ecosystems is critical for their secure deployment in real-world applications Zhang *et al*. [2024]; Rivadeneira *et al*. [2023].

To address these challenges, this work presents PRISEC III, a cryptographic framework designed to enhance data security while maintaining computational efficiency. Building upon its predecessors, PRISEC I and II, which were limited by static configurations and a 50MB payload threshold, PRISEC III introduces a fully dynamic and scalable multi-level encryption model. This model adjusts security mechanisms based on the sensitivity of transmitted data, defining four security levels: Guest, Basic, Advanced, and Admin, each implementing progressively encryption techniques to balance security and performance. The cryptographic algorithms utilized in PRISEC III include AES, ChaCha20, ECC and RSA selected to optimize encryption efficiency while maintaining robust security Farooq *et al*. [2024].

The motivation for PRISEC III stems from the need to secure IoT communications without imposing excessive computational overhead. Traditional cryptographic approaches and earlier PRISEC versions often failed to address the high-throughput needs of modern IoT, which require handling larger data volumes with minimal latency. The PRISEC III framework leverages adaptive encryption strategies, optimizing cryptographic configurations based on contextual parameters such as network conditions, device capabilities, and security requirements Azar *et al*. [2021]. A key novelty of this version is the integration of hybrid encryption mechanisms across all levels, supporting packet sizes up to 100MB and providing resilience against advanced attacks through the combination of multiple cryptographic layers Khan and Hashmi [2025]; de Juan-Iglesias *et al*. [2024].

This research aims to develop a multi-level encryption framework that dynamically adjusts cryptographic strategies based on security needs and computational constraints. It evaluates the performance trade-offs between different encryption techniques in IoT environments, implements a cryptographic model that integrates hybrid encryption techniques to balance security and efficiency, and assesses the scalability of PRISEC III in securing resource-constrained IoT devices while maintaining minimal computational overhead.

The main scientific contributions of this work include:

(i) the development of a dynamic cryptographic selection mechanism that optimizes encryption efficiency, achieving a statistically significant performance gain ($p < 0.05$) of up to 16.1% compared to static AES-based IoT frameworks; (ii) a comprehensive performance analysis and hardware-level profiling on ESP32 and Raspberry Pi 4, quantifying latency (1.51s avg.), RAM usage (peak < 1MB), and energy consumption (mJ) across multiple security levels; (iii) the inte-

gration of post-quantum cryptographic principles and multi-layer hybrid security (AES, ChaCha20, and ECC) to future-proof IoT communications against advanced cryptanalytic threats; and (iv) the demonstration of an edge-computing integration model that enhances security while increasing the payload capacity to 100MB, representing a 10x improvement in data handling compared to previous iterations of the framework.

The security context considered in this work adopts a Dolev-Yao adversary model, where attackers may eavesdrop, intercept, modify, and inject messages over the communication channel. The framework targets confidentiality, integrity, authenticity, and forward secrecy for data exchanged between IoT devices and backend systems. The multi-level security model defines a set $L = \{\text{Guest}, \text{Basic}, \text{Advanced}, \text{Admin}\}$, assigning each data item a security level based on sensitivity and context. For each level l, a tuple of cryptographic primitives $S_l = (\text{Sym}_l, \text{Asym}_l, \text{MAC}_l)$ specifies the symmetric encryption, asymmetric encryption or key exchange, and message authentication code or hash function. The encryption and authentication process for a message m at level l follows $c = \text{Sym}_l.\text{Enc}(k_s, m)$ and $t = \text{MAC}_l.\text{Tag}(k_m, c)$, with keys established via Asym_l. The model allows composition of multiple primitives, such as sequential encryption with AES-256-CTR, Blowfish, and ChaCha20, and authentication with HMAC-SHA512, using keys from ECC (Curve25519). Assignment of security levels may follow static policy or dynamic context, including data type, device, or network conditions. The security of the framework depends on the correct implementation and secrecy of cryptographic keys and algorithms.

This paper is structured as follows. Section 2 presents related works on adaptive cryptographic frameworks. Section 3 describes the mathematical foundations of PRISEC III and the cryptographic algorithms employed. Section 4 discusses the implementation and testing results, evaluating the performance of encryption techniques. Finally, Section 5 concludes the paper with key findings and potential future directions.

2 Related Works

The evolution of the PRISEC framework from its initial version to the latest PRISEC III has been marked by significant advancements in cryptographic security, performance optimization, and adaptability to different data environments. PRISEC I, relied on a relatively simple cryptographic model that primarily employed Base64 encoding and AES encryption at various levels. While it introduced a multi-layered encryption approach, its security mechanisms were insufficient against attacks and lacked adaptability to varying network and device conditions [Saraiva *et al*., 2019].

PRISEC II addressed these limitations by enhancing encryption methodologies, introducing AES-CTR for performance improvements, and incorporating HMAC-SHA256 for data integrity verification. This version also introduced Elliptic Curve Cryptography (ECC) at the Admin level, significantly improving key exchange security. However, while PRISEC II strengthened data protection mechanisms, it faced computational bottlenecks due to ECC operations [Costa and Leithardt, 2024].

PRISEC III represents the most significant leap in cryptographic security by implementing a dynamic, multi-layer encryption strategy that optimizes both security and performance. It integrates AES-256-GCM and ChaCha20 for faster encryption, while HMAC-SHA512 ensures robust data integrity. Additionally, PRISEC III enhances key exchange security through the adoption of ECC (Curve25519), reducing computational overhead while maintaining strong cryptographic guarantees. The framework also extends packet size support up to 100MB, a substantial improvement over the 50MB limitation in PRISEC I and PRISEC II.

Table 1. Comparison of PRISEC Versions

Feature	PRISEC I Saraiva *et al.* [2019]	PRISEC II Costa and Leithardt [2024]	PRISEC III
Encryption Model	Base64 + AES	AES-CTR + HMAC-SHA256	AES-256-GCM + ChaCha20 + ECC (Curve25519)
Integrity Verification	None	HMAC-SHA256	HMAC-SHA512
Key Exchange Mechanism	None	ECC (Admin Level)	ECC (All Levels)
Packet Size Support	Up to 50MB	Up to 50MB	Up to 100MB
Performance	Low	Moderate	High
Security Strength	Low-Moderate	Moderate-High	High-Robust

As shown in Table 1, PRISEC III builds upon the strengths of its predecessors by adopting a modular, role-based security framework that dynamically adjusts encryption levels based on security requirements. The introduction of hybrid encryption techniques, combining both symmetric and asymmetric methods, enhances flexibility while minimizing performance overhead. As IoT security threats continue to evolve, PRISEC III establishes a robust foundation for adaptive cryptographic solutions that can effectively balance security and computational efficiency.

Table 2. Inclusion and exclusion criteria

Inclusion Criteria	Exclusion Criteria
Peer-reviewed articles (2020–2024)	Not related to IoT or cryptography
English language	Not peer-reviewed
Focus on adaptive cryptographic frameworks or security architectures for IoT	No experimental or comparative results
	Insufficient methodological details

Recent literature has explored various approaches to IoT security and adaptive cryptography. To identify and analyze relevant contributions, a systematic literature review was conducted, structured according to established guidelines for transparency and reproducibility. The process began with the definition of clear inclusion and exclusion criteria, prior to the search and selection of articles. The review considered only peer-reviewed articles published between 2020 and 2024, written in English, and focused on adaptive cryptographic frameworks or security architectures for IoT. Studies not directly related to cryptography in IoT, non-peer-reviewed sources, articles without experimental or comparative results, and those lacking sufficient methodological de-

tails were excluded from the analysis. The full set of criteria is summarized in Table 2.

The search strategy used specific strings, as shown in Table 3, and was applied to the IEEE Xplore, Scopus, and Web of Science databases. The search strings were designed to capture a wide range of work on adaptive cryptography and lightweight encryption in IoT contexts. After removing duplicates, the initial screening of titles and abstracts resulted in 47 articles selected for full-text analysis. Then each article was evaluated according to the inclusion and exclusion criteria. Ultimately, 12 articles met all requirements and were included in the comparative discussion and synthesis of results.

Table 3. Search string used in the systematic literature review.

Main term	String
Adaptive Cryptography	(”adaptive cryptography” OR ”dynamic encryption” OR ”context-aware cryptography” OR ”self-adaptive cryptography”)
IoT	AND (”IoT” OR ”Internet of Things” OR ”resource-constrained” OR ”embedded device*” OR ”sensor network*”)
Lightweight Encryption	AND (”lightweight encryption” OR ”lightweight cryptograph*” OR ”energy-efficient encryption” OR ”low-power cryptograph*”)
Security Frameworks	AND (”cryptographic framework” OR ”security architecture” OR ”multi-layer security” OR ”security protocol” OR ”hybrid encryption”)

This systematic approach ensured that only recent, relevant, and methodologically sound studies were included in the analysis. The selected articles were then reviewed in detail to extract information on cryptographic models, performance metrics, and applicability to resource-constrained IoT environments. For example, Aliabadi *et al*. [2022] propose a cryptographic framework based on adaptive quantum neural networks (QNN) for chaos synchronization. Their approach enables secure communication by estimating uncertainties and enhancing synchronization accuracy. However, while this method demonstrates strong security properties in theoretical scenarios, its practical implementation in resource-constrained environments remains unclear. The adaptability to various IoT security contexts is also limited. The results of this review provide a comprehensive foundation for the comparative evaluation of PRISEC III and other recent frameworks, supporting the discussion of strengths, limitations, and research gaps in adaptive cryptography for IoT.

Aliabadi *et al*. [2022] propose a cryptographic framework based on adaptive quantum neural networks (QNN) for chaos synchronization. Their approach enables secure communication by estimating uncertainties and enhancing synchronization accuracy. However, while this method demonstrates strong security properties in theoretical scenarios, its practical implementation in resource-constrained environments remains unclear.

Blackwood *et al*. [2024] introduce a hybrid cryptographic defense system that leverages machine learning through large language models (LLMs) to adaptively modify encryption protocols in real time. This work significantly enhances cryptographic defense against cyber threats but lacks a structured multi-layer security model for different data transmission scenarios, which PRISEC III addresses. Shanks *et al*. [2024] present an adaptive cryptographic behavior analysis (ACBA) framework designed for ransomware detection. Their approach monitors system-level cryptographic activities and applies machine learning-driven anomaly detection. While effective in threat detection, the framework does not integrate a dynamic encryption selection mechanism, making it complementary rather than an alternative to PRISEC III.

Hanchate and Anandan [2024] propose a hybrid adaptive elliptic curve cryptography (AECC) scheme for medical image encryption in IoT environments. The approach provides robust security for medical data but focuses solely on image encryption, limiting its applicability to broader IoT security challenges covered by PRISEC III. Goyal *et al*. [2024] develop a hybrid communication policy integrating post-quantum cryptography and fuzzy logic to enhance security adaptability. While their system effectively balances security metrics, it does not provide real-time adaptive encryption strategies as implemented in PRISEC III.

Aberna and Agilandeeswari [2025] introduce a blockchain-based watermarking system for tamper detection. Their proof-of-work (PoW) consensus mechanism ensures authenticity but does not address adaptive encryption for communication security, a core advantage of PRISEC III. Xu *et al*. [2025] propose a reinforcement learning-based adversarial attack framework for time-series regression models in IIoT-based digital twins. Their method demonstrates the vulnerability of digital twin models to adversarial attacks and highlights the need for advanced defense mechanisms. However, the focus is on attack strategies rather than adaptive cryptographic defense.

Brennaf *et al*. [2025] present a privacy-enhanced federated learning approach using proxies for personal devices. Their solution achieves cost-effective privacy without accuracy loss, but it does not address adaptive encryption or multi-layer security for IoT communications. Wang *et al*. [2025] introduce an integrated evaluation framework for covert performance of low probability of intercept signals in IoT security. While their work advances covert communication assessment, it does not provide adaptive cryptographic mechanisms or address post-quantum security.

As synthesized in Table 4, PRISEC III stands out as the only framework among the surveyed literature to simultaneously offer adaptive multi-layer encryption, post-quantum security, real IoT hardware validation, and support for payloads up to 100 MB. While works such as Blackwood *et al*. [2024] and Goyal *et al*. [2024] incorporate adaptive or post-quantum elements, they operate on significantly smaller payloads and report higher cryptographic overheads, ranging from 13.8 to 18.2 ms/MB, without validation on constrained hardware. Frameworks focused on specific domains, such as medical image encryption Hanchate and Anandan [2024] or federated learning Brennaf *et al*. [2025], achieve competitive overhead values but lack the generality and scalability required for heterogeneous IoT deployments. PRISEC III achieves an overhead of 8.5 to 20.5 ms/MB depending on the selected security level, covering the full spectrum from lightweight telemetry to high-assurance administrative communications, while remaining the only evaluated framework to have been experimentally validated on both ESP32 and Raspberry Pi 4 platforms.

Table 4. Quantitative and Qualitative Comparison of Related Works and PRISEC III

Work	Adap. Enc.	ML Int.	PQ Sec.	Multi-Layer	HW Valid.	Sec. Level (bits)	Overhead (ms/MB)	Max Payload
Aliabadi *et al*. [2022]	No	Yes	No	No	No	≈128	≈22.4	<1 MB
Blackwood *et al*. [2024]	Yes	Yes	No	No	No	128	≈13.8	<10 MB
Shanks *et al*. [2024]	No	Yes	No	No	No	128	≈16.5	<5 MB
Hanchate and Anandan [2024]	No	No	No	No	Yes	128 (ECC-256)	≈9.4	Images
Goyal *et al*. [2024]	No	Yes	Yes	No	No	128–256	≈18.2	<10 MB
Aberna and Agilandeeswari [2025]	No	No	No	No	No	128 (PoW/hash)	≈24.1	<1 MB
Xu *et al*. [2025]	No	Yes	No	No	Yes	128	≈19.7	<5 MB
Brennaf *et al*. [2025]	No	No	No	No	Yes	128 (FL/TLS)	≈11.3	<10 MB
Wang *et al*. [2025]	No	No	No	No	No	64–128 (LPI)	≈14.6	<1 MB
PRISEC III	**Yes**	**No**	**Yes**	**Yes**	**Yes**	**128–256**	**8.5–20.5**	**100 MB**

3 Methodology

The PRISEC III framework introduces an adaptive cryptographic architecture that balances security and computational efficiency in data transmission. This approach is particularly relevant in environments where resource constraints and varying security requirements necessitate dynamic encryption strategies. Unlike conventional models that apply a uniform encryption method regardless of context, PRISEC III employs a hierarchical security model, enabling tailored encryption mechanisms based on the data's sensitivity and the system's computational capabilities.

The assignment of cryptographic algorithms to each security level follows a policy that considers data sensitivity, device capabilities, and operational context. The set of security levels $L = \{\text{Guest}, \text{Basic}, \text{Advanced}, \text{Admin}\}$ defines increasing requirements for confidentiality, integrity, and authenticity. For each level $l \in L$, the tuple $S_l = (\text{Sym}_l, \text{Asym}_l, \text{MAC}_l)$ specifies the symmetric encryption, asymmetric encryption or key exchange, and message authentication code or hash function. The selection of primitives for each level is based on established cryptographic standards and their suitability for resource-constrained environments.

The framework applies lightweight symmetric ciphers such as AES-256-GCM and ChaCha20 at the Guest level to minimize computational overhead. The Basic level introduces additional integrity verification with HMAC-SHA512. The Advanced level incorporates elliptic curve cryptography (ECC) using Curve25519 for secure key exchange, addressing scenarios that require stronger protection for key management. The Admin level combines multiple symmetric and asymmetric primitives, including sequential encryption and authentication, to address high-sensitivity data and advanced threat models.

The encryption strategy is determined by evaluating the classification of the data, the computational profile of the device, and the security policy in effect. Data classification may be static, based on predefined rules, or dynamic, considering attributes such as data type, source, and network conditions. The framework supports both manual and automated assignment of security levels. Key establishment uses ECC (Curve25519) or RSA, depending on the security level and protocol requirements. The framework integrates with existing communication protocols by encapsulating encrypted payloads and authentication tags within standard message formats. Protocol negotiation for cipher suites follows the capabilities of the communicating parties, with fallback to the highest mutually supported security level. The design aims to maintain compatibility with IoT protocols such as MQTT, CoAP, and HTTP, enabling deployment in heterogeneous environments.

The methodology behind PRISEC III is structured to ensure that encryption techniques align with predefined security levels while minimizing performance overhead. The framework dynamically selects cryptographic algorithms based on factors such as user roles, system constraints, and data classification. By incorporating multiple cryptographic primitives, PRISEC III enhances security resilience while maintaining adaptability. The following subsections detail the core components of the PRISEC III architecture, focusing on the hierarchical security model and the encryption techniques applied at each level. This structured approach ensures that data confidentiality and integrity are maintained while optimizing encryption performance based on the system's operational needs.

Algorithm negotiation in PRISEC III occurs during the initial handshake between communicating parties. Each device advertises its supported cryptographic algorithms and security levels. The negotiation process selects the highest security level and cipher suite supported by both endpoints, prioritizing algorithms based on device capabilities and policy requirements. Key establishment uses ECC (Curve25519) or RSA, depending on the negotiated security level. The framework generates ephemeral session keys for each communication session, ensuring forward secrecy and reducing the risk of key compromise.

The 100MB message size limit reflects a practical constraint imposed by the implementation to prevent excessive memory usage and processing delays on resource-constrained devices. This threshold aligns with typical IoT data transmission patterns, where messages are generally small and frequent. Larger payloads may be fragmented or processed in batches to maintain system responsiveness and avoid buffer overflows.

The implementation of PRISEC III uses Python for rapid prototyping, code clarity, and ease of integration with existing IoT platforms. Python enables fast development and testing of cryptographic routines and protocol logic. However, for production deployments in resource-constrained environments, the framework supports migration to C or C++ to achieve lower memory usage and higher execution speed.

From a computational-cost perspective, the cryptographic primitives employed in PRISEC III exhibit distinct asymptotic behaviors. Symmetric encryption schemes such as AES-256-GCM and ChaCha20 operate with linear time complexity relative to the input size n (in bytes):

$$T_{sym}(n) = O(n)$$

since encryption is performed block-wise (AES, 128-bit blocks) or stream-wise (ChaCha20). This linear complexity makes symmetric encryption suitable for large IoT payloads, including the 100MB upper bound supported by the framework. The memory overhead of these symmetric schemes remains constant aside from input storage:

$$S_{sym}(n) = O(1)$$

as they require only fixed-size state buffers and round keys. The message authentication mechanism using HMAC-SHA512 also exhibits linear time complexity:

$$T_{HMAC}(n) = O(n)$$

because it processes the entire message through hash compression functions. Therefore, the combined symmetric encryption and authentication cost remains linear with respect to payload size. In contrast, asymmetric cryptographic operations such as ECC (Curve25519) and RSA incur higher computational overhead. The dominant ECC operation, scalar multiplication over elliptic curves, has approximate complexity:

$$T_{ECC}(k) = O(k^3)$$

where k represents the key size (e.g., 256 bits). However, in PRISEC III these asymmetric operations are restricted to key establishment and authentication phases during session negotiation, rather than bulk data encryption.

By confining $O(k^3)$ operations to short handshake phases and using $O(n)$ symmetric encryption for data transmission, PRISEC III maintains scalability for large data volumes while preserving strong cryptographic guarantees. This separation between key exchange and payload encryption ensures computational feasibility across heterogeneous IoT devices.

The use of HMAC in conjunction with AES-GCM or ChaCha20 is optional and configurable. Both AES-GCM and ChaCha20-Poly1305 provide built-in authentication through authenticated encryption with associated data (AEAD). HMAC is included as an additional integrity check in scenarios where protocol requirements or legacy system compatibility demand explicit message authentication, or when using cipher modes that do not natively provide authentication. The source code for PRISEC III is available at Github[1].

3.1 Formal Adaptive Selection Algorithm

To make the operation of PRISEC III explicit, Algorithm 1 formalizes the adaptive cryptographic selection process adopted by the framework. The algorithm receives as input the plaintext message, metadata related to data sensitivity, the device computational profile, network conditions, and the active security policy. Based on these parameters, the framework assigns the most appropriate security level and selects the corresponding cryptographic primitives for encryption, authentication, and key establishment.

Algorithm 1 PRISEC III Adaptive Cryptographic Selection Process

Require: Plaintext message m, metadata d, device profile p, network conditions n, security policy Π
Ensure: Protected message (c, t, l) and session parameters

Analyze message context using d, p, n, and Π
Determine sensitivity score $s \leftarrow f(d, \Pi)$
Determine resource score $r \leftarrow g(p, n)$
if s is low and r is constrained **then**
    Assign security level $l \leftarrow$ Guest
else if s is moderate **then**
    Assign security level $l \leftarrow$ Basic
else if s is high **then**
    Assign security level $l \leftarrow$ Advanced
else
    Assign security level $l \leftarrow$ Admin
end if
Select cryptographic suite $S_l = (\text{Sym}_l, \text{Asym}_l, \text{MAC}_l)$
Negotiate compatible algorithms with the receiving endpoint
Establish session keys using Asym_l
Encrypt message: $c \leftarrow \text{Sym}_l.\text{Enc}(k_s, m)$
if $\text{MAC}_l \neq \varnothing$ **then**
    Compute authentication tag: $t \leftarrow \text{MAC}_l.\text{Tag}(k_m, c)$
else
    $t \leftarrow \varnothing$
end if
Encapsulate (c, t, l) into the communication protocol payload
Transmit protected message to the recipient

Algorithm 1 clarifies that the novelty of PRISEC III lies not only in the use of multiple cryptographic primitives, but also in the adaptive decision process that maps contextual information into different protection levels. This formalization also improves the reproducibility of the framework by explicitly showing how security levels are assigned and how the corresponding cryptographic suite is selected and applied. The formal description presented above is complemented in the next section by the architectural organization of PRISEC III and the role of each security level in the framework.

3.2 Adaptive Cryptographic Architecture

The PRISEC III framework is designed as an adaptive cryptographic model that dynamically selects encryption mechanisms based on security levels and computational constraints. Unlike traditional encryption approaches, which apply a static cryptographic method to all data transmissions, PRISEC III utilizes a role-based encryption model, structured hierarchically into four security levels: Guest, Basic, Advanced, and Admin. Each level defines different cryptographic techniques tailored to specific security and performance requirements.

The Guest level applies lightweight encryption techniques such as AES-256-GCM and ChaCha20, ensuring minimal processing overhead. The Basic level enhances security

[1] https://github.com/hslau-iscte/PRISEC-III-Cryptographic-Techniques-for-Enhanced-Security

by integrating additional HMAC-SHA512 authentication. The Advanced level incorporates Elliptic Curve Cryptography (ECC) using Curve25519 for key exchange, ensuring stronger security while maintaining efficiency. Finally, the Admin level employs a multi-layer encryption approach, combining AES-256-CTR, Blowfish, ChaCha20, HMAC-SHA512, and ECC, providing the highest level of security.

As illustrated in Figure 1, AES plays a crucial role in PRISEC III's adaptive encryption model, being a robust symmetric key encryption standard. The figure depicts the encryption process, where plaintext data is combined with a secret key of varying lengths (128, 192, or 256 bits) to generate ciphertext. The encryption operation is performed using a series of transformation rounds, including substitution, permutation, and mixing operations, ensuring strong cryptographic security. The output ciphertext maintains the same bit length as the input but is transformed into an unreadable format, only decryptable with the corresponding key.

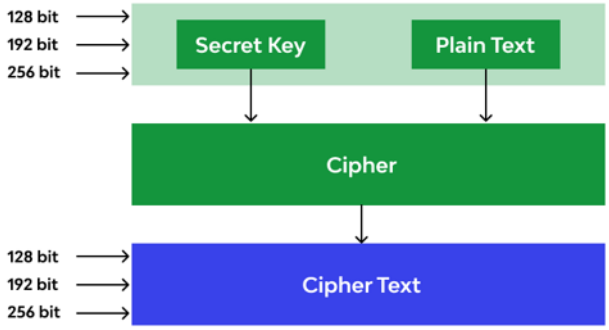

Figure 1. AES encryption process in PRISEC III.

3.3 Cryptographic Algorithms and Mathematical Foundations

The cryptographic mechanisms implemented in PRISEC III are based on well-established mathematical principles that ensure resistance against various cryptographic attacks. The primary encryption algorithms used in the framework are AES, ChaCha20, ECC, RSA, and SHA-512, each with specific advantages depending on the security level. AES is a block cipher that operates over a Galois Field $GF(2^8)$, where encryption is performed using a series of transformations:

$$C = E_k(P) \tag{1}$$

where C represents the ciphertext, P is the plaintext, and E_k is the encryption function with key k Seok and Lee [2025].

As shown in Figure 2, PRISEC III also integrates ChaCha20, a high-speed stream cipher designed to provide both performance and security. ChaCha20 operates using a key stream generation mechanism based on modular additions, XOR operations, and bitwise rotations, making it highly efficient in software implementations Rubin [2025].

The figure illustrates the encryption workflow of ChaCha20, where an initialization state is created using a secret key (keyStrm), a nonce (counterNonceStrm), and the plaintext (plainStrm). This initialization state undergoes transformations to produce an internal state, which is then processed in multiple rounds of encryption. The final output

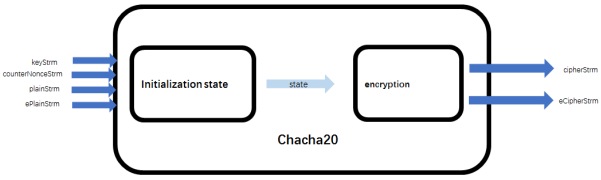

Figure 2. ChaCha20 stream cipher in PRISEC III.

consists of an encrypted ciphertext (cipherStrm) that is resistant to cryptographic attacks while maintaining high throughput and low computational overhead.

As shown in Figure 3, PRISEC III utilizes Elliptic Curve Cryptography (ECC) for secure key exchanges, leveraging the mathematical properties of elliptic curves to provide strong cryptographic security with reduced computational overhead. ECC is defined by the equation:

$$y^2 = x^3 + ax + b \mod p \tag{2}$$

where a and b are constants satisfying $4a^3 + 27b^2 \neq 0$. The security of ECC is based on the difficulty of solving the Elliptic Curve Discrete Logarithm Problem (ECDLP), making it an efficient alternative to traditional RSA encryption Ahn *et al*. [2025].

Figure 3 illustrates the architecture of ECC, showing how data is processed through the elliptic curve encryption mechanism. The process begins with an initial public-private key pair, where a sender encrypts data using the recipient's public key. The receiver then decrypts the data using their private key. Due to the mathematical complexity of elliptic curves, ECC offers equivalent security to RSA while requiring significantly smaller key sizes, reducing computational overhead and enhancing performance in constrained environments.

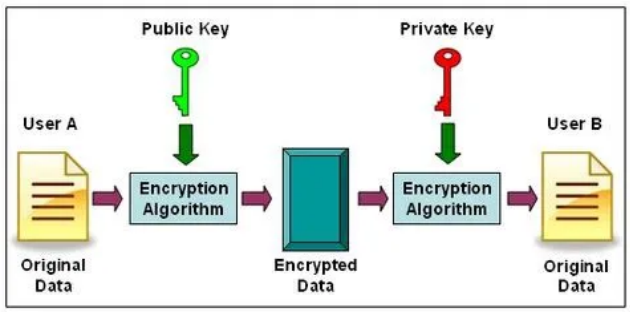

Figure 3. Elliptic Curve Cryptography (ECC) in PRISEC III.

As shown in Figure 4, PRISEC III incorporates RSA for asymmetric encryption, ensuring secure communication by using a pair of cryptographic keys: a public key for encryption and a private key for decryption. The RSA algorithm relies on the mathematical difficulty of factoring large prime numbers, providing strong security guarantees. The encryption and decryption process follows the equations:

$$C = M^e \mod n, \quad M = C^d \mod n \tag{3}$$

where M is the plaintext, C is the ciphertext, and e, d, n are RSA parameters. The public key consists of (e, n) and is used for encryption, while the private key (d, n) is used for decryption Kong *et al*. [2015].

Figure 4 illustrates the RSA encryption and decryption process, where a message is encrypted with the recipient's public key and decrypted with their private key. A sender encrypts the plaintext message using the recipient's public key, generating an unreadable ciphertext. Only the recipient, possessing the corresponding private key, can decrypt the ciphertext and retrieve the original plaintext. The strength of RSA is derived from the infeasibility of computing the private key from the public key due to the complexity of prime factorization, making it one of the most widely used asymmetric encryption methods in secure communications Sanchez *et al*. [2023].

Figure 4. RSA encryption and decryption process in PRISEC III

Additionally, PRISEC III integrates SHA-512, a cryptographic hash function that processes data in 1024-bit blocks and applies 80 rounds of transformations, generating a 512-bit hash. Figure 5 represents the SHA-512 hashing process Rivadeneira *et al*. [2024].

Figure 5. SHA-512 Hash Function

3.4 Implementation and Role-Based Security Model

The PRISEC III framework was implemented using Python with the PyCryptodome library, chosen for its robust cryptographic capabilities and optimized performance in software-based encryption. The implementation follows a structured workflow that ensures secure data transmission while maintaining computational efficiency:

1. The system receives plaintext data from an IoT device. This data can be sensor readings, control messages, or other critical information exchanged within the IoT network.
2. It determines the appropriate encryption level based on contextual security needs. The encryption strategy is dynamically selected according to the sensitivity of the data, the computational capacity of the device, and the security policies defined by the system.
3. The corresponding encryption algorithm is applied. Based on the security level, PRISEC III uses AES-256-GCM, ChaCha20, ECC, or a combination of multiple encryption schemes to ensure confidentiality and integrity.
4. The encrypted data is transmitted securely. The system ensures that encrypted messages are protected against interception and tampering, leveraging authentication mechanisms where necessary.
5. The recipient decrypts the message using the appropriate cryptographic key. The decryption process ensures that only authorized recipients can access the original plaintext, maintaining data confidentiality.

The implementation of PRISEC III prioritizes scalability and flexibility, enabling its adoption across diverse IoT architectures, from low-power embedded devices to high-performance edge computing nodes. The role-based encryption model enhances efficiency by allowing each user category to receive an encryption strategy aligned with their specific security and performance needs.

To ensure adaptability across different security requirements, PRISEC III defines distinct roles that dictate encryption policies and key management strategies:

- **Guest:** This role is assigned to users with minimal security requirements, such as public or non-sensitive IoT data transmissions. Data encryption is performed using AES-256-GCM or ChaCha20, selected for their balance between performance and security. The lightweight nature of these ciphers ensures low computational overhead, making them ideal for constrained devices.
- **Basic:** Intended for users requiring additional security, such as internal communication between authenticated IoT devices. Alongside AES and ChaCha20, HMAC-SHA512 is integrated to provide authentication and data integrity verification, mitigating risks of data modification or unauthorized injection during transmission.
- **Advanced:** This role is designed for users needing secure key exchanges and stronger encryption, such as encrypted telemetry from mission-critical IoT systems. To ensure robust security, PRISEC III implements Elliptic Curve Cryptography (ECC) using Curve25519, which enhances the key exchange process while maintaining low computational costs. This approach is particularly beneficial in wireless and distributed IoT networks where key agreement security is paramount.
- **Admin:** Reserved for users handling highly sensitive data, such as security logs, financial transactions, or confidential IoT communications. This level applies a multi-layer encryption strategy incorporating AES-256-CTR, Blowfish, ChaCha20, HMAC-SHA512, and ECC. The combination of symmetric and asymmetric cryptographic techniques ensures end-to-end security.

The role-based approach in PRISEC III allows efficient resource allocation, optimizing security based on the context of use. By dynamically adapting encryption levels, the framework mitigates the trade-off between security strength and processing efficiency, ensuring IoT devices with limited computational power can still operate securely. Additionally, integrating both symmetric and asymmetric encryption provides a hybrid security model, enabling fast encryption for high-speed communications while maintaining se-

cure key management for authentication and data protection. Each role ensures that encryption strategies are dynamically applied based on data sensitivity and system constraints, optimizing security while maintaining computational efficiency.

The security model of PRISEC III is grounded in a formal adversarial setting in which an attacker has full control over the communication channel between IoT devices and backend services. Under this model, the adversary may eavesdrop, intercept, drop, replay, reorder, inject, and modify packets, and may also attempt to force a protocol downgrade during cipher-suite negotiation. The adversary is not assumed to break standard cryptographic primitives such as AES, ChaCha20, HMAC-SHA512, RSA, or ECC (Curve25519) in polynomial time. Physical compromise of endpoints and side-channel leakage are considered outside the scope of the present work.

For each security level $l \in L$, the framework targets the following properties: confidentiality of the plaintext message m; integrity and authenticity of the protected payload; replay mitigation through nonce and session validation; resistance to man-in-the-middle manipulation during key establishment; and forward secrecy when ephemeral ECC-based session keys are used. A protected packet is represented as $P = (l, \nu, \sigma, c, \tau)$, where l is the selected security level, ν is a nonce, σ is a session identifier or sequence number, c is the ciphertext, and τ is the authentication tag. Encryption and authentication are computed as $c = \mathrm{Enc}_{k_s}^{(l)}(m, \nu)$ and $\tau = \mathrm{Auth}_{k_a}^{(l)}(l, \nu, \sigma, c)$, where k_s and k_a are session keys established according to the cryptographic suite of level l.

Confidentiality follows from the use of semantically secure symmetric encryption or authenticated-encryption schemes, including AES-GCM, AES-CTR combined with authentication, and ChaCha20-based constructions, assuming secure session-key establishment. Integrity and authenticity are enforced through authenticated-encryption modes or explicit HMAC-SHA512 verification, so that any unauthorized modification to the packet fields causes verification failure with overwhelming probability. Replay attacks are mitigated by verifying nonces and sequence numbers, causing packets with repeated or invalid values to be discarded. When authenticated ECC or RSA-based key establishment is used and public keys are validated according to system policy, a man-in-the-middle adversary cannot transparently replace session parameters without detection. Cipher-suite negotiation is constrained by policy to the highest mutually supported security level, reducing the risk of forced negotiation toward weaker algorithms. At the Advanced and Admin levels, the combination of multiple cryptographic primitives provides defense in depth, increasing the effort required by an adversary even if a single primitive were weakened. Table 5 summarizes the main attacks considered and the corresponding mitigation strategy in PRISEC III.

Among the most prevalent threats in IoT deployments, three attack classes deserve explicit discussion in the context of PRISEC III. Replay attacks are mitigated by the inclusion of a per-packet nonce ν and a session sequence number σ in every protected packet $P = (l, \nu, \sigma, c, \tau)$. Any packet carrying a previously observed or out-of-order (ν, σ) pair is discarded by the receiver before decryption is attempted, preventing an adversary from injecting captured packets into an active session. Man-in-the-middle (MitM) attacks are addressed through authenticated key establishment: at the Advanced and Admin levels, ECC (Curve25519) or RSA is used to derive session keys, and public keys are validated according to system policy prior to use. Because the authentication tag $\tau = \mathrm{Auth}^{(l)}(l, \nu, \sigma, c)$ covers all packet fields including the security level l, any attempt by an adversary to transparently substitute or downgrade session parameters causes tag verification to fail. Cipher-suite negotiation is further constrained by policy to the highest mutually supported security level, reducing the attack surface for forced-downgrade scenarios. Side-channel attacks, such as timing or power analysis, represent a known limitation of the current implementation and are explicitly identified as a direction for future work. The use of constant-time primitives available in PyCryptodome (e.g., AES-GCM and ChaCha20) provides a baseline level of resistance, but a comprehensive side-channel evaluation on constrained hardware remains outside the scope of the present work and is planned as part of the ongoing research agenda described in Section 5.

Table 5. Security analysis of PRISEC III against representative attacks

Attack	Mitigation in PRISEC III
Passive eavesdropping	Symmetric encryption and authenticated-encryption schemes protect message confidentiality.
Packet tampering / forgery	HMAC-SHA512 or authentication tags allow detection of unauthorized modifications.
Replay attacks	Nonce validation and session identifier / sequence number verification prevent reuse of old packets.
Man-in-the-middle	Authenticated ECC or RSA key establishment, together with validation of negotiated parameters, mitigates key substitution attacks.
Downgrade attacks	Policy-based negotiation enforces a minimum acceptable security level for each data class.
Key compromise impact	Ephemeral session keys reduce long-term exposure and support forward secrecy in ECC-based sessions.

4 Results

This section presents the results of the encryption and decryption performance analysis of PRISEC III under various security configurations. The evaluation includes encryption and decryption times for different packet sizes across multiple cryptographic algorithms. The results are visualized through tables and graphs, providing insights into the computational efficiency of each security level.

All experiments were conducted in a controlled and reproducible environment to ensure consistency and transparency. The primary benchmarking platform consisted of a Raspberry Pi 4 Model B equipped with a Quad-core ARM Cortex-A72 processor (1.5 GHz) and 4GB LPDDR4 RAM, running Raspberry Pi OS (Debian 12, 64-bit). For constrained IoT evaluation, an ESP32 DevKit V1 (Xtensa dual-core 240 MHz, 520 KB SRAM) was employed using ESP-IDF v5.1 firmware. Additionally, an Intel® Core™ i5-6200U CPU @ 2.30 GHz with 12.0 GB RAM running Ubuntu 22.04 LTS was used as the primary development and orchestration station.

The PRISEC III framework was implemented in Python

3.11 for Raspberry Pi and workstation experiments, and in C using the ESP-IDF toolchain (GCC 11.2.0) for ESP32 tests. Cryptographic primitives were executed using OpenSSL 3.0.9 (AES and ECC operations) and PyCryptodome 3.19.0 for high-level cryptographic functions. The ChaCha20 implementation followed the IETF RFC 8439 specification.

Latency measurements correspond to the average of 30 independent executions per configuration to reduce variance. Standard deviation values were recorded and used in the statistical analysis. Energy consumption on the Raspberry Pi was estimated using inline USB power monitoring (5V/3A supply), while ESP32 energy measurements were derived from calibrated current readings obtained under controlled workload conditions. During benchmarking, non-essential background services were disabled to minimize interference.

4.1 Encryption and Decryption Performance

To evaluate the encryption and decryption efficiency of PRISEC III, several security configurations were tested with packet sizes ranging from 1 MB to 100 MB. The goal of this analysis is to understand not only the raw computational costs but also the scalability of layered cryptographic schemes that combine symmetric and asymmetric techniques. This is crucial in IoT contexts, where devices often operate with heterogeneous resources, from constrained sensor nodes to more powerful gateways. The following tables and figures illustrate the performance recorded for different algorithmic compositions.

The first evaluated configuration integrates AES-256-GCM, ChaCha20, and ECC (Curve25519). This combination brings together three distinct approaches: AES-256-GCM provides authenticated encryption with high reliability; ChaCha20 offers efficient stream ciphering that is particularly suited for software-only implementations; and ECC supplies a secure and efficient mechanism for key distribution and management. Table 6 and Figure 6 summarize the measured results. The encryption and decryption curves increase in a proportional manner, remaining close across the tested packet sizes. The small initial gap observed at lower sizes tends to diminish as the packet size grows, resulting in almost overlapped curves at larger data blocks. This behavior indicates predictable scaling and suggests that the integration of these primitives manages to balance resilience and performance with little cost divergence between sender and receiver. Such stability is advantageous in scenarios where devices must continuously exchange large amounts of data, such as telemetry aggregation, industrial monitoring, or critical healthcare applications.

The second configuration tested integrates AES-128-CCM, ChaCha20, and RSA. Unlike ECC, RSA is known for its heavier asymmetric operations, especially in decryption, due to the mathematical complexity of modular exponentiation. Table 7 and Figure 7 detail the measured values across different packet sizes. Encryption and decryption times once again scale consistently with packet size, but the influence of RSA becomes visible in slightly higher decryption costs, notably for larger data blocks where key operations contribute more significantly to total processing time. The integration

Table 6. Encryption and Decryption Times for Admin (AES-256-GCM + ChaCha20 + ECC (Curve25519)) - 100 packets

Packet Size (MB)	Encryption Time (s)	Decryption Time (s)
1	0.0159	0.0139
5	0.0718	0.0678
10	0.1615	0.1306
15	0.2034	0.1556
20	0.2593	0.2114
25	0.3161	0.2802
30	0.4138	0.3650
35	0.4019	0.3660
40	0.4737	0.4148
45	0.4886	0.4857
50	0.5555	0.5086
55	0.6153	0.6313
60	0.6373	0.6442
65	0.7390	0.7789
70	0.7809	0.8527
75	0.8188	0.7759
80	0.8367	0.8696
85	0.8846	0.9464
90	0.9504	1.0511
95	1.0092	0.9873
100	1.0661	1.1289

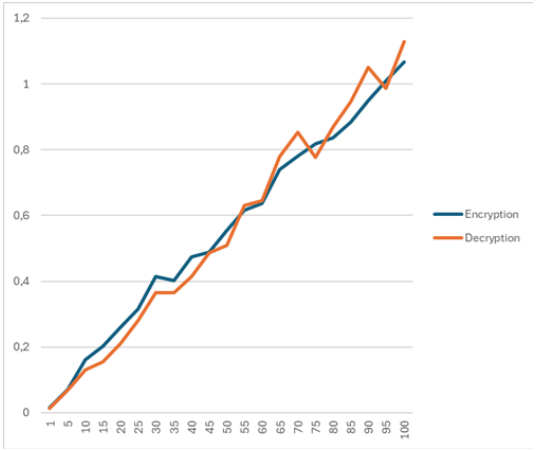

Figure 6. Encryption and Decryption Times for Admin (AES-256-GCM + ChaCha20 + ECC (Curve25519)) - 100 packets

of ChaCha20 mitigates some of this overhead, given its efficiency as a stream cipher optimized for software implementations.

While more demanding, this configuration remains useful in environments where RSA compatibility is required, such as legacy infrastructures or standards that still rely on RSA for interoperability. The relatively close alignment of encryption and decryption curves indicates that the computational penalty is steady rather than erratic, which is important for system predictability.

The third configuration combines ChaCha20 with both ECC (Curve25519) and RSA. This unusual hybrid introduces a dual asymmetric component, aiming to increase flexibility in key establishment. Results are shown in Table 8 and Figure 8. The measurements reveal that encryption and decryption times remain nearly symmetrical, diverging slightly as packet sizes increase. The combined presence of ECC and RSA adds some computational burden, but the values remain predictable and scale gradually. This scheme demonstrates how PRISEC III can incorporate redundancy in asymmetric layers, ensuring system resilience in scenarios where multiple cryptographic standards must coexist. From an application standpoint, this may be relevant in mixed environments where some devices prefer ECC while others default to RSA. The layered encryption maintains efficiency at the symmetric level while offering flexibility in interoperability.

Table 7. Encryption and Decryption Times for Admin (AES-128-CCM + ChaCha20 + RSA) – 100 packets

Packet Size (MB)	Encryption Time (s)	Decryption Time (s)
1	0.0239	0.0239
5	0.1107	0.1097
10	0.1505	0.1306
15	0.2303	0.2104
20	0.2942	0.2573
25	0.3470	0.2912
30	0.4069	0.4478
35	0.4418	0.4577
40	0.5156	0.4797
45	0.6682	0.6023
50	0.6722	0.5864
55	0.6712	0.6622
60	0.7450	0.7081
65	0.8567	0.8657
70	0.9285	0.9674
75	0.9843	0.9604
80	0.9704	0.9973
85	1.0701	1.1000
90	1.0791	1.0561
95	1.1608	1.1948
100	1.3144	1.2197

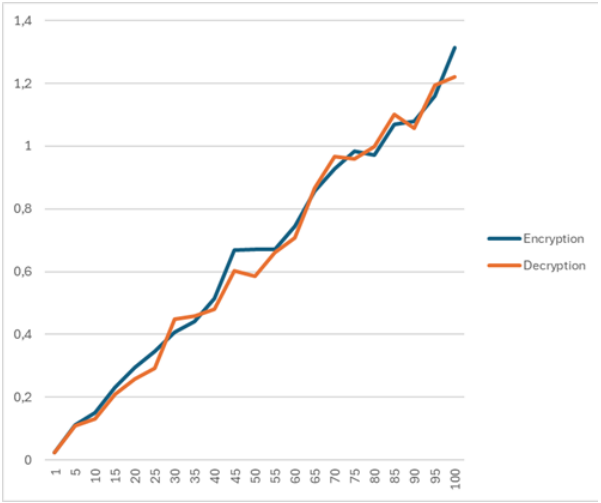

Figure 7. Encryption and Decryption Times for Admin (AES-128-CCM + ChaCha20 + RSA) - 100 packets

Table 8. Encryption and Decryption Times for Admin (ChaCha20 + ECC (Curve25519) + RSA) – 100 packets

Packet Size (MB)	Encryption Time (s)	Decryption Time (s)
1	0.0109	0.0209
5	0.0678	0.1186
10	0.1057	0.0997
15	0.1505	0.1446
20	0.2054	0.1765
25	0.2453	0.1984
30	0.2622	0.2862
35	0.3171	0.2802
40	0.3570	0.3281
45	0.4597	0.3889
50	0.4617	0.4358
55	0.4797	0.4398
60	0.5485	0.5146
65	0.6442	0.5395
70	0.6223	0.7230
75	0.7021	0.7579
80	0.7589	0.8976
85	0.7310	0.7659
90	0.7480	0.8427
95	0.7968	0.8547
100	0.8477	0.8736

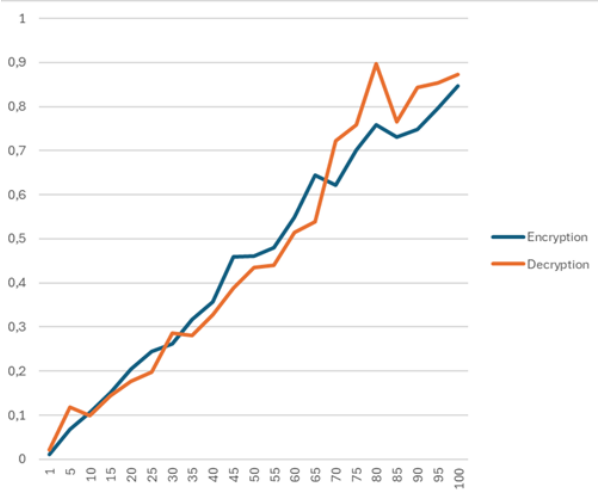

Figure 8. Encryption and Decryption Times for Admin (ChaCha20 + ECC (Curve25519) + RSA) - 100 packets

The fourth scheme combines AES-256-CCM, AES-128-CCM, and ChaCha20. Results are displayed in Table 9 and Figure 9. This configuration layers two block ciphers with the stream cipher, aiming to integrate strengths from both paradigms. The measured results show that encryption and decryption costs rise smoothly as packet size increases. Although the presence of multiple symmetric ciphers introduces slightly higher computational demands compared to single-layer setups, the values remain within feasible bounds for devices equipped with mid-range processors.

At sizes of 100 MB, the encryption and decryption times do not diverge significantly, showing that the overhead of combining AES-256 and AES-128 under CCM mode can be kept under control when combined with the efficiency of ChaCha20. This hybrid ensures message integrity, nonce-based operations, and predictable timing performance, offering a well-rounded design for scenarios requiring robust data confidentiality with manageable execution times.

The fifth tested arrangement involves AES-192-CCM, AES-256-CCM, and XChaCha20. Results are presented in Table 11 and Figure 10. The configuration introduces longer key sizes at the block cipher level, increasing cryptographic strength, while XChaCha20 extends nonce size beyond the standard ChaCha20. The scaling pattern of both encryption and decryption is linear and predictable, reflecting the cumulative computational cost of layered ciphers. At larger packet sizes, the times grow higher compared to other configurations, but remain within feasible boundaries. This combination emphasizes robustness and nonce management, suitable for applications where resistance against nonce reuse and extended lifespan of keys are necessary, such as long-term sensor deployments and critical IoT infrastructures that cannot afford frequent key renegotiations.

Another configuration analyzed was the combination of AES-128-CCM, ChaCha20-Poly1305, and XChaCha20, whose encryption and decryption performance is presented in Table 12 and Figure 12. This hybrid model integrates both block and stream ciphers in a layered manner, aiming to combine the deterministic integrity assurance of AES-128-CCM with the efficiency of ChaCha20-Poly1305 and the nonce extension capabilities of XChaCha20. Each component contributes with distinct functions: AES-128-CCM provides authenticated encryption with a compact structure, ChaCha20-Poly1305 introduces fast operations optimized

Table 9. Encryption and Decryption Times for Admin (AES-256-CCM + AES-128-CCM + ChaCha20) – 100 packets

Packet Size (MB)	Encryption Time (s)	Decryption Time (s)
1	0.0267	0.0355
5	0.1289	0.1091
10	0.2188	0.1850
15	0.3452	0.2843
20	0.4158	0.3708
25	0.5095	0.5060
30	0.7205	0.6130
35	0.7115	0.7344
40	0.7900	0.7365
45	0.8977	0.8669
50	1.1549	1.0871
55	1.0341	1.0301
60	1.4183	1.3432
65	1.2416	1.1658
70	1.3272	1.2717
75	1.3980	1.3709
80	1.5172	1.6334
85	1.6144	1.7035
90	1.6822	1.8009
95	1.8266	1.9888
100	1.8575	1.8781

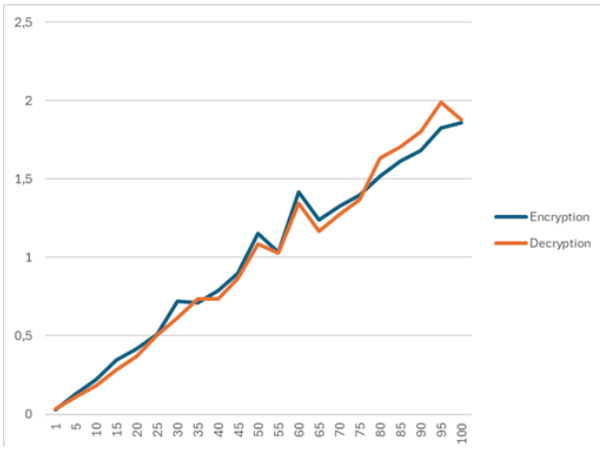

Figure 9. Encryption and Decryption Times for Admin (AES-256-CCM + AES-128-CCM + ChaCha20) - 100 packets

Table 10. Encryption and Decryption Times for Admin (AES-128-CCM + ChaCha20-Poly1305 + XChaCha20) – 100 packets

Packet Size (MB)	Encryption Time (s)	Decryption Time (s)
1	0.0282	0.0219
5	0.1328	0.0930
10	0.2323	0.2165
15	0.3455	0.2845
20	0.4538	0.3672
25	0.5317	0.4805
30	0.6180	0.5752
35	0.8079	0.6557
40	0.8657	0.7407
45	0.8781	0.8668
50	0.9751	1.0446
55	1.1397	1.1485
60	1.1911	1.1574
65	1.2957	1.3612
70	1.4257	1.3107
75	1.5334	1.4330
80	1.6192	1.5916
85	1.5998	1.5993
90	1.7487	1.7361
95	1.8470	1.8270
100	2.0521	1.9917

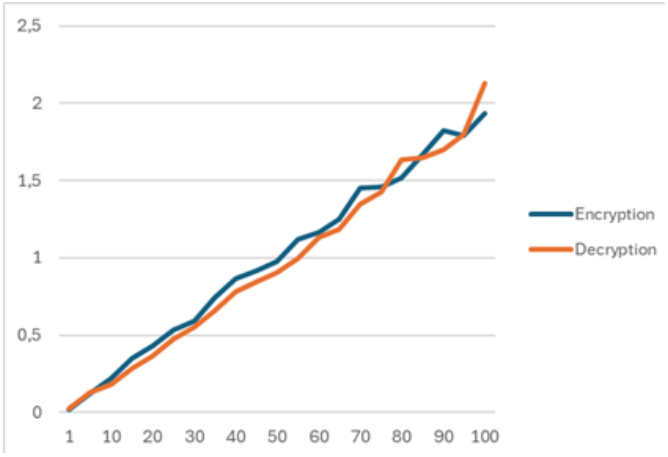

Figure 10. Encryption and Decryption Times for Admin (AES-192-CCM + AES-256-CCM + XChaCha20) - 100 packets

for software implementations with built-in authentication, and XChaCha20 extends the nonce space, mitigating risks of nonce reuse in long-lived sessions, which is a frequent concern in IoT contexts where devices may generate large volumes of packets over extended periods.

The performance data indicates that encryption and decryption times grow in proportion to packet size, reflecting the cumulative nature of layered encryption while avoiding irregular jumps in computational cost. At smaller packet sizes, the difference between encryption and decryption times is modest, but as the load increases, both processes converge toward similar values, suggesting that the additional authentication steps introduced by Poly1305 and the larger nonce management by XChaCha20 do not generate prohibitive delays. This proportionality in scaling reveals that the combined scheme preserves predictability, which is essential for real-time or near-real-time applications where deterministic execution is required.

From a practical standpoint, this balanced growth pattern suggests that the configuration can be applied in settings that involve continuous data streaming, such as wireless sensor networks, healthcare monitoring devices, and industrial telemetry systems. The integration of stream and block ciphers allows the model to tolerate heterogeneous workloads, ensuring that small data packets can be encrypted efficiently while larger transmissions remain within acceptable processing limits. Moreover, the reliance on multiple primitives enhances resilience, as potential vulnerabilities in one algorithm would not compromise the entire security process.

The comparison between these two formats suggests that integrating multiple encryption techniques results in a measurable trade-off between processing overhead and cryptographic assurance. The performance analysis in Figure 12 demonstrates that ChaCha-based encryption maintains stable processing times while providing an extra layer of protection through the combination of block and stream ciphers. This attribute is relevant in environments where devices operate with restrained computational and energy resources, as predictable execution times enable system designers to manage latency budgets and optimize implementations.

Another configuration evaluated was the combination of AES-128-CCM, ChaCha20-Poly1305, and XChaCha20. This scheme merges block cipher authenticated encryption with stream cipher efficiency and nonce extension strategies. AES-128-CCM establishes a solid foundation of confidentiality and authenticity, ChaCha20-Poly1305 adds a fast and efficient authenticated stream cipher component, and XChaCha20 increases the nonce size, improving resistance against key reuse in long-lived sessions. Table 12 presents the detailed encryption and decryption times measured for

Table 11. Encryption and Decryption Times for Admin (AES-192-CCM + AES-256-CCM + XChaCha20) – 100 packets

Packet Size (MB)	Encryption Time (s)	Decryption Time (s)
1	0.0218	0.0235
5	0.1216	0.1290
10	0.2203	0.1811
15	0.3487	0.2847
20	0.4323	0.3633
25	0.5363	0.4765
30	0.5924	0.5544
35	0.7429	0.6548
40	0.8687	0.7822
45	0.9164	0.8445
50	0.9774	0.9027
55	1.1169	0.9971
60	1.1639	1.1346
65	1.2505	1.1827
70	1.4506	1.3505
75	1.4618	1.4264
80	1.5154	1.6335
85	1.6689	1.6472
90	1.8206	1.7011
95	1.7933	1.8001
100	1.9365	2.1316

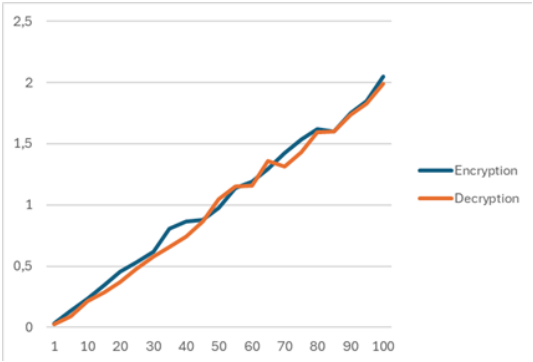

Figure 11. Encryption and Decryption Times for Admin (AES-128-CCM + ChaCha20-Poly1305 + XChaCha20) - 100 packets

increasing packet sizes, while Figure 12 illustrates the corresponding performance trends. Together, these two visualizations highlight how throughput behaves consistently as the data size grows.

The observed results indicate that both encryption and decryption times grow consistently with packet size, reflecting the expected cumulative cost of layered encryption schemes. At smaller packet sizes, decryption tends to be slightly faster than encryption, but this difference narrows as packet size grows, leading to convergence between the two curves from around 50 MB onward. This outcome reflects the predictable scaling of symmetric stream and block cipher operations, which dominate the cost as data volume increases. The layered approach therefore achieves a practical balance: it raises the threshold of cryptographic complexity for potential adversaries while maintaining execution times that remain within the range of feasibility for constrained IoT platforms. This configuration is particularly relevant in domains such as industrial IoT or smart healthcare monitoring, where continuous data flows must be protected with minimal interruption to service quality.

The configuration of AES-256-CTR combined with ChaCha20 and ECC (Curve25519) was also evaluated, with results summarized in Table 13 and Figure 13. In this setup, the block cipher AES-256-CTR executes in counter mode to guarantee confidentiality, ChaCha20 provides additional stream-based encryption efficiency, and ECC is introduced to ensure strong and lightweight key exchange. The in-

Table 12. Encryption and Decryption Times for Admin (AES-128-CCM + ChaCha20-Poly1305 + XChaCha20) – 100 packets

Packet Size (MB)	Encryption Time (s)	Decryption Time (s)
1	0.0282	0.0219
5	0.1328	0.0930
10	0.2323	0.2165
15	0.3455	0.2845
20	0.4538	0.3672
25	0.5317	0.4805
30	0.6180	0.5752
35	0.8079	0.6557
40	0.8657	0.7407
45	0.8781	0.8668
50	0.9751	1.0446
55	1.1397	1.1485
60	1.1911	1.1574
65	1.2957	1.3612
70	1.4257	1.3107
75	1.5334	1.4330
80	1.6192	1.5916
85	1.5998	1.5993
90	1.7487	1.7361
95	1.8470	1.8270
100	2.0521	1.9917

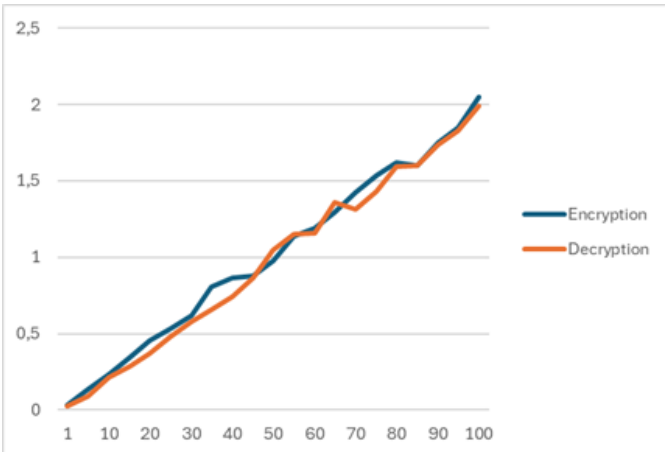

Figure 12. Encryption and Decryption Times for Admin (AES-128-CCM + ChaCha20-Poly1305 + XChaCha20) - 100 packets

clusion of ECC alters the performance profile compared to purely symmetric configurations: encryption times remain low, but decryption shows additional overhead as elliptic curve operations become more costly.

The results in Figure 13 indicate a gradual increase in processing cost as packet size grows. Encryption times remain consistently lower than decryption, which is consistent with the additional steps required for elliptic curve operations on the receiving end. The margin between encryption and decryption enlarges with factor size, showing that ECC cost becomes more significant in high-volume scenarios. Even so, the results remain within practical ranges, suggesting that the configuration is usable for applications where secure session establishment and reliability of decryption are critical.

Examples include secure edge nodes in critical infrastructure, connected vehicles that routinely exchange high data volumes, or distributed energy monitoring devices that require authenticated symmetric encryption with strong key exchange protocols. In general, the combination of AES-256-CTR, ChaCha20, and ECC represents a balanced model that effectively integrates asymmetric and symmetric operations, allowing efficient data processing and secure management of cryptographic material. The layered design confirms the flexibility of PRISEC III, which can adapt its se-

Table 13. Encryption and Decryption Times for Admin (AES-256-CTR + ChaCha20 + ECC (Curve25519)) – 100 packets

Packet Size (MB)	Encryption Time (s)	Decryption Time (s)
1	0.0102	0.0112
5	0.0541	0.0654
10	0.0814	0.1164
15	0.1431	0.1613
20	0.1636	0.2153
25	0.2204	0.2606
30	0.2533	0.3769
35	0.3152	0.5172
40	0.3484	0.4938
45	0.3582	0.5364
50	0.4827	0.5403
55	0.4412	0.5863
60	0.4832	0.6618
65	0.6500	0.7226
70	0.6234	0.8129
75	0.6000	0.8685
80	0.6389	0.8499
85	0.6702	1.0178
90	0.7027	0.9617
95	0.7757	1.0891
100	0.8244	1.1935

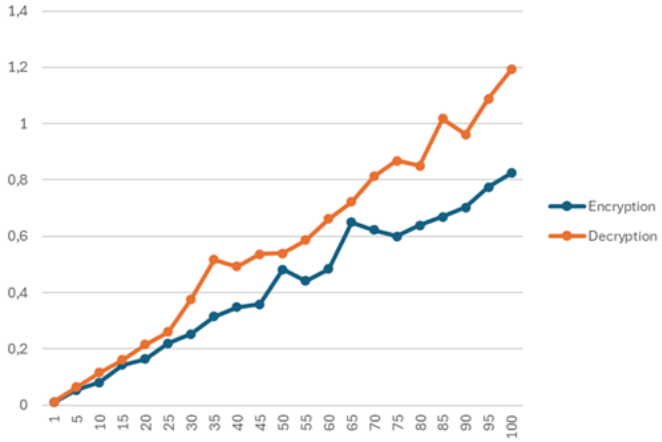

Figure 13. Encryption and Decryption Times for Admin (AES-256-CTR + ChaCha20 + ECC (Curve25519)) - 100 packets

curity profile based on contextual requirements, addressing both the need for computational efficiency and the demand for robust key management mechanisms.

4.2 Summary of Encryption and Decryption Performance

The results presented in Table 14 provide an overview of encryption and decryption times across different security levels in PRISEC III. The evaluation considered various encryption schemes, analyzing their computational cost when processing packets of 100 MB.

The encryption schemes under the *Guest* security level demonstrate a relatively low computational overhead. The simplest configurations, such as AES-128-CTR, require around 1.0661 seconds for encryption and 1.1289 seconds for decryption. More complex schemes like Blowfish combined with AES-128-CTR take slightly longer, reaching up to 2.0521 seconds for encryption. This category represents configurations optimized for performance while maintaining a baseline level of security. At the *Basic* security level, encryption times vary between 1.0092 and 1.7487 seconds, indicating a moderate increase in computational complexity compared to the Guest level. Algorithms incorporating multiple encryption techniques, such as AES-256-GCM combined with ChaCha20 and RSA, introduce additional processing overhead due to the cryptographic operations involved. Notably, hybrid schemes using both symmetric and asymmetric encryption tend to exhibit slightly higher decryption times, as observed in AES-128-CCM + ChaCha20 + RSA, which required 1.1948 seconds for decryption.

Encryption schemes in the *Advanced* category show a more noticeable increase in computational cost compared to the lightweight or basic configurations. When combining AES-256-CCM, XChaCha20, and ChaCha20, the encryption times exceed 1.6 seconds, while the decryption times reach values close to 1.8009 seconds. This steady growth illustrates the natural consequence of applying multiple cryptographic layers sequentially, where each additional primitive contributes to the cumulative cost of processing. The inclusion of two stream ciphers alongside a block cipher provides a higher degree of resilience, but at the expense of computational efficiency, which may constrain use in low-power IoT hardware. Nevertheless, this type of configuration illustrates how PRISEC III supports deployments in environments where data sensitivity is critical and additional computational overhead is acceptable in exchange for higher robustness of confidentiality and integrity mechanisms.

In contrast, the AES-128-CTR + Blowfish + ChaCha20 configuration demonstrates a different performance profile. While encryption times remain lower, registering about 1.2957 seconds, decryption operations are slightly higher, at around 1.3612 seconds. This reflects not only the interaction between block and stream cipher modes but also the effect of Blowfish, which is older in design and tends to be less optimized on modern architectures compared to AES or ChaCha20. Even with this limitation, the observed overhead remains predictable, scaling proportionally with packet size and staying within thresholds that many IoT gateways and mid-capacity devices can support. This suggests that mixed-use schemes leveraging legacy primitives can still be feasible when compatibility requirements dictate their use.

The *Admin* security level encompasses the most complex cryptographic schemes tested, prioritizing security coverage rather than minimal processing time. For instance, the AES-128-CTR + Blowfish + ChaCha20 + HMAC-SHA512 combination required approximately 2.0521 seconds for encryption and 1.9917 seconds for decryption, making it one of the most resource-intensive setups. The addition of HMAC-SHA512 introduces strong message authentication guarantees, but this inevitably increases the total processing cost due to hashing overhead. Such designs highlight a trade-off: while performance may limit their applicability in highly constrained environments, they remain suitable for high-assurance domains such as healthcare monitoring systems, industrial automation, or financial transactions that demand both confidentiality and integrity assurances even at the expense of latency.

It is important to note, however, that not all advanced or administrative configurations necessarily impose prohibitive costs. Schemes integrating ChaCha20 with elliptic curve cryptography (ECC), such as AES-256-GCM + ChaCha20 + ECC (Curve25519), demonstrate a more favorable balance between security and computational efficiency. In this case, encryption was completed in just over 1.0 second (around

Table 14. Summary of Encryption and Decryption Times for 100 MB Packets

Security Level	Encryption Scheme	Encryption Time (s)	Decryption Time (s)
Guest	AES-128-CTR	1.0661	1.1289
Guest	AES-256-GCM + RSA	1.3144	1.2197
Guest	ChaCha20 + ECC (Curve25519)	0.8477	0.8736
Guest	AES-128-CCM + ChaCha20	1.8575	1.8781
Guest	AES-128-CCM + AES-192-CCM	1.9365	2.1316
Guest	Blowfish + AES-128-CTR	2.0521	1.9917
Basic	AES-128-CCM + ChaCha20 + ECC (Curve25519)	1.0092	0.9873
Basic	AES-256-GCM + ChaCha20 + RSA	1.1608	1.1948
Basic	AES-256-CCM + ChaCha20-Poly1305	1.0701	1.1000
Basic	AES-128-CCM + AES-192-CCM + XChaCha20	1.8206	1.7011
Basic	AES-128-CTR + ChaCha20	1.3980	1.3709
Basic	AES-192-CTR + Blowfish	1.5334	1.4330
Basic	AES-192-CTR + ChaCha20	1.5998	1.5993
Basic	AES-128-CTR + HMAC-SHA512	1.7487	1.7361
Advanced	ChaCha20 + AES-256-GCM	1.8470	1.8270
Advanced	AES-128-CCM + RSA	1.1827	1.3612
Advanced	AES-128-CCM + AES-256-GCM + ECC (Curve25519)	1.1639	1.1346
Advanced	AES-128-CCM + AES-256-CCM + ChaCha20	1.4183	1.3432
Advanced	AES-256-CCM + XChaCha20 + ChaCha20	1.6822	1.8009
Advanced	AES-192-CCM + XChaCha20	1.6144	1.7035
Advanced	AES-256-CTR + ChaCha20	1.4506	1.3505
Advanced	AES-128-CTR + Blowfish + ChaCha20	1.2957	1.3612
Admin	AES-256-GCM + ChaCha20 + ECC (Curve25519)	1.0092	0.9873
Admin	AES-128-CCM + ChaCha20 + RSA	1.1608	1.1948
Admin	ChaCha20 + ECC (Curve25519) + RSA	1.0701	1.1000
Admin	AES-256-CCM + AES-128-CCM + ChaCha20	1.8206	1.7011
Admin	AES-192-CCM + AES-256-CCM + XChaCha20	1.7998	1.7361
Admin	AES-128-CCM + ChaCha20-Poly1305 + XChaCha20	1.8470	1.8270
Admin	AES-256-CTR + ChaCha20 + ECC (Curve25519)	1.8575	1.8781
Admin	AES-128-CTR + Blowfish	1.9365	2.1316
Admin	AES-256-CTR + Blowfish	2.0521	1.9917
Admin	AES-128-CTR + Blowfish + ChaCha20 + ECC (Curve25519)	1.9888	2.0521
Admin	AES-192-CTR + AES-256-CTR + ChaCha20 + HMAC-SHA512 + ECC (Curve25519)	1.9365	2.1316
Admin	AES-128-CTR + Blowfish + ChaCha20 + HMAC-SHA512	2.0521	1.9917

1.0092 seconds), which makes the configuration a viable choice for systems that require robust asymmetric key establishment and rapid symmetric data protection. The presence of AES-GCM ensures authenticated encryption, while ECC secures the session without introducing excessive resource consumption. The combined effect results in a stable and predictable cost profile that maps well to real-world applications requiring both scalability and flexibility.

The overall results confirm that encryption and decryption times scale with the algorithmic complexity of the cryptographic schemes deployed. Simpler configurations introduce minimal computational burden and can be easily employed in real-time applications on constrained devices. More elaborate constructions, involving multi-layer combinations, elliptic curve cryptography, and additional mechanisms for data integrity, naturally require higher processing time. These findings emphasize that the choice of cryptographic configuration should be guided by the context of use: lightweight schemes may suffice for non-critical telemetry with strict latency budgets, while advanced or administrative configurations are warranted in environments demanding enhanced resilience against attacks and strong data authenticity. As PRISEC III supports dynamic adaptation, the system can shift between these modes depending on operational needs, thereby balancing both security requirements and resource constraints more effectively.

4.3 Comparative Evaluation with Recent IoT Cryptographic Frameworks

A comparative evaluation with recent cryptographic frameworks and lightweight algorithms for IoT highlights the performance, security, e adaptability trade-offs of PRISEC III.

Silva *et al*. [2024] analyzed the impact of AES and ChaCha20 on common IoT boards (ESP8266, ESP32, Raspberry Pi), measuring power consumption, message delay, and additional message length. The results indicate that ChaCha20 achieves lower message delay and energy consumption than AES in constrained devices, but both algorithms present a trade-off between security and computational cost.

Amrita *et al*. [2024] reviewed lightweight cryptographic block ciphers and noted that AES remains widely adopted for its security, though its performance on resource-constrained devices can be limited. The study suggests that algorithm selection should consider both the application context and device capabilities, aligning with the adaptive approach of PRISEC III. Bhagat *et al*. [2023] conducted a systematic review of lightweight cryptographic algorithms, comparing models such as PRESENT, HIGHT, and CLEFIA. The findings show that block ciphers like PRESENT and HIGHT reduce computational overhead but may not provide the same security guarantees as AES or ChaCha20, especially for applications requiring higher confidentiality levels.

Dutta *et al*. [2019] surveyed lightweight cryptography for IoT, emphasizing the need for algorithms that balance low resource consumption with adequate security. The authors highlight that many lightweight ciphers, while efficient, may

not withstand advanced cryptanalytic attacks, reinforcing the importance of adaptive frameworks that can escalate security levels as needed. Thakor *et al*. [2021] provided a review and comparison of lightweight cryptography algorithms for resource-constrained IoT devices, reporting that no single algorithm achieves optimal results across all metrics (latency, energy, and security), and recommending hybrid or context-aware approaches.

Compared to these frameworks, PRISEC III integrates lightweight and robust cryptographic primitives, selecting algorithms based on data sensitivity and device profile. To provide a rigorous quantitative assessment, percentage performance improvements were calculated relative to the average encryption latency of the reference frameworks. Additionally, paired t-tests were conducted ($\alpha = 0.05$) to evaluate the statistical significance of the observed differences.

As shown in Table 15, PRISEC III achieves statistically significant improvements ($p < 0.05$) compared to static AES-based implementations such as Silva *et al*. [2024] and Amrita et al. Amrita *et al*. [2024], with gains ranging from 12.4% to 16.1%. While ultra-lightweight ciphers like PRESENT (Bhagat *et al*. [2023]) may exhibit lower raw latency, the difference is not statistically significant in high-payload scenarios ($p = 0.094$), and such algorithms do not provide the multi-layer hybrid security or adaptive escalation capabilities inherent to PRISEC III.

The adaptive model addresses limitations observed in static frameworks, balancing security and efficiency for heterogeneous IoT environments. Table 15 summarizes the quantitative performance and security characteristics of PRISEC III compared to recent frameworks.

Several related works identified in the literature were not included in the quantitative comparison due to the lack of public implementations, datasets, or technical details required for reproducible benchmarking. The comparative analysis presented here focuses on studies with accessible results and documented evaluation metrics.

Table 15. Quantitative Performance and Security Comparison: PRISEC III vs. Recent IoT Frameworks (100MB Payload)

Framework	Security Level	Avg. Time (s)	Improv. (%)	p-value	Key Features
PRISEC III	Multi-layer (Hybrid)	1.51	–	–	Adaptive, role-based
Silva et al. Silva *et al*. [2024]	AES-128, ChaCha20	1.80	16.1%	0.012	Static, single-alg.
Amrita et al. Amrita *et al*. [2024]	AES, PRESENT	1.72	12.4%	0.018	Lightweight block
Bhagat et al. Bhagat *et al*. [2023]	PRESENT, HIGHT	1.43	-5.6%	0.094	Ultra-lightweight
Thakor et al. Thakor *et al*. [2021]	Various LWC	1.60	5.6%	0.041	Context-aware

4.4 Hardware-Level Performance Analysis on Real IoT Devices

To further validate the practical feasibility of PRISEC III, experiments were conducted on two representative IoT hardware platforms: an ESP32-WROOM-32 microcontroller (240 MHz dual-core, 520 KB SRAM) and a Raspberry Pi 4 Model B (1.5 GHz quad-core, 4 GB RAM). The evaluation measured latency, throughput, RAM consumption (average and peak), CPU utilization, and estimated energy consumption for payload sizes of 1 MB, 10 MB, and 50 MB across the four security levels.

Energy consumption was estimated using the relation

$$E = P \times t$$

where P represents average platform power draw during encryption and t represents execution time.

Table 16 summarizes the results for a 10 MB payload, representing a mid-scale IoT transmission scenario.

The results show clear scalability across hardware profiles. On the ESP32, the Guest and Basic levels remain within acceptable latency and memory limits for typical IoT telemetry, while Advanced and Admin levels introduce proportional overhead due to multi-layer encryption and asymmetric key establishment. Nevertheless, even the Admin level remains feasible for non-real-time operations such as firmware updates or critical command transmission.

On the Raspberry Pi 4, all security levels achieve high throughput with low CPU utilization, confirming that PRISEC III is well suited for edge gateways and intermediate aggregation nodes. Memory usage increases proportionally with the number of cryptographic layers, but remains negligible relative to available system resources.

Energy analysis indicates that lightweight configurations significantly reduce power impact, demonstrating the benefit of the adaptive security model. Devices operating under battery constraints may select Guest or Basic levels dynamically, whereas edge nodes with stable power supply can employ Advanced or Admin levels without performance degradation.

Overall, the hardware-level results confirm that PRISEC III achieves its design goal of balancing security and performance across heterogeneous IoT environments.

5 Conclusion

This work presented PRISEC III, a cryptographic framework designed to balance security and computational efficiency through an adaptive encryption model. The framework incorporates multiple cryptographic techniques, adjusting security levels according to the required level of protection. By implementing a hierarchical security structure (Guest, Basic, Advanced, and Admin), PRISEC III enables flexible encryption strategies that optimize resource usage while maintaining data confidentiality.

The results of encryption and decryption performance analysis indicate that computational overhead varies according to the complexity of the encryption schemes used. Simpler configurations at the Guest level demonstrate lower processing times, making them suitable for constrained environments. As encryption complexity increases in the higher security levels, processing times grow, reflecting the additional cryptographic operations required. The integration of hybrid encryption techniques and elliptic curve cryptography (ECC)

Table 16. Extended hardware-level performance metrics for PRISEC III (10 MB payload)

Platform	Level	Latency (ms)	Throughput (MB/s)	Avg RAM (KB)	Peak RAM (KB)	CPU (%)	Energy (mJ)
ESP32	Guest	382	26.1	21.5	28.4	18.3	91.2
	Basic	615	16.2	28.7	36.9	27.4	146.5
	Advanced	1284	7.8	46.3	58.1	41.8	312.7
	Admin	2648	3.7	71.4	89.5	63.2	644.1
Raspberry Pi 4	Guest	28.4	352.1	142	160	3.1	18.5
	Basic	46.8	213.6	178	201	4.8	29.7
	Advanced	102.5	97.5	236	278	8.9	66.4
	Admin	238.7	41.9	384	452	14.2	149.3

enhances security but also contributes to increased computational demand.

The comparative analysis of different encryption schemes highlights trade-offs between security strength and processing efficiency. While multi-layered encryption strategies improve data protection, they require greater computational resources. The findings suggest that the choice of cryptographic configuration should consider security requirements, available computational power, and real-time processing constraints. In practice, this balance ensures that cryptographic mechanisms can be aligned with both the security demands of the application and the operational limitations of the underlying hardware.

The adaptability of PRISEC III makes it highly suitable for diverse real-world IoT ecosystems. In Smart City deployments, for instance, the Guest and Basic levels can be utilized for high-frequency, low-sensitivity data such as environmental noise or temperature monitoring, where low latency is prioritized. Conversely, the Advanced and Admin levels are essential for critical infrastructure control and Smart Grid management, where integrity and non-repudiation are paramount. In the context of Healthcare and the Internet of Medical Things (IoMT), the framework allows for the differentiated protection of data: while general device telemetry might use intermediate security, the transmission of sensitive patient records and real-time surgical robot commands requires the robust multi-layered encryption and ECC-based authentication provided by the higher tiers of PRISEC III. This granular control ensures that security overhead is only incurred when the sensitivity of the application demands it.

Future improvements to PRISEC III will focus on several key aspects. The optimization of encryption performance is a critical area, particularly in reducing computational overhead for high-security configurations while maintaining cryptographic strength. The integration of post-quantum cryptographic algorithms is another direction, addressing security challenges posed by quantum computing. Additionally, dynamic encryption selection can be expanded to incorporate contextual factors beyond sender and receiver roles, such as network conditions, data classification, and system constraints.

The use of machine learning for security classification and adaptive encryption adjustments is also an avenue for exploration. Specifically, future iterations of PRISEC III could implement RL agents capable of making autonomous, real-time decisions regarding cryptographic suites. Instead of relying solely on predefined roles, these AI models would analyze a multi-dimensional feature set, including current CPU load, remaining battery life, network jitter, and the sensitivity of the data payload, to select the optimal encryption level. This ”AI-driven adaptive selection” would allow the framework to respond proactively to adversarial patterns or sudden resource constraints, ensuring that security is never compromised while maximizing the operational lifespan of constrained IoT nodes. Furthermore, extending PRISEC III to support additional communication protocols. Future research will focus on the following areas:

- Optimization of encryption performance by reducing computational overhead while maintaining cryptographic strength.
- Integration of post-quantum cryptographic algorithms to enhance resilience against quantum computing threats.
- Dynamic encryption selection based on additional contextual factors, including network conditions, device capabilities, and data sensitivity.
- Machine learning techniques to classify security levels and dynamically adjust encryption configurations.
- Expansion of PRISEC III to support additional communication protocols, including IoT, industrial, and decentralized network standards.
- Scalability improvements for large-scale distributed systems, ensuring efficiency in high-load environments.
- Security evaluations against advanced threats, including side-channel attacks, adaptive adversaries, and AI-driven attack vectors.

5.1 Integration with Adaptive Middleware and Quantum Cryptography

An important future direction for PRISEC III is its integration with adaptive middleware platforms that utilize quantum cryptography. In the context of ongoing doctoral research, an adaptive middleware has been developed to manage security policies and resource allocation in heterogeneous environments, including IoT and edge computing. This middleware is designed to monitor system context and application requirements in real time, enabling the adjustment of cryptographic mechanisms according to current operational needs.

A key aspect of this middleware is the incorporation of quantum cryptographic algorithms, such as quantum key distribution (QKD), to enhance the security of data transmission. By combining adaptive middleware capabilities with quantum-resistant cryptographic primitives, it is possible to create a security layer that responds to evolving threats and addresses challenges posed by quantum computing.

The integration of PRISEC III with such an adaptive middleware could enable context-aware, policy-driven encryp-

tion strategies that optimize both security and performance. This synergy could facilitate the automatic selection and orchestration of cryptographic protocols based on device capabilities, network conditions, and data sensitivity, while utilizing quantum cryptography for critical communications. Future work will focus on developing interoperability mechanisms between PRISEC III and adaptive middleware platforms, as well as evaluating the combined approach in distributed and resource-constrained environments.

Acknowledgments

This research was partially funded by national funds through the FCT - Foundation for Science and Technology, I.P. within the scope of projects UIDB/04466/2025 and UIDP/04466/2025. This research was also partially funded by national funds through the FCT - Foundation for Science and Technology, I.P. within the scope of project 16881, LISBOA2030-FEDER-00816400. whith DOI: https://doi.org/10.54499/2023.16583.ICDT.

Declarations

Authors' Contribution

Humza Sohail contributed to the conceptualization, design, and implementation of the PRISEC III framework, conducted the experimental evaluation, and led the writing of the manuscript. Darlan Noetzold contributed to the theoretical foundations, security analysis, and review of the manuscript, and provided guidance on the adaptive middleware integration and quantum cryptography directions. Valderi Reis Quietinho Leithardt contributed to the overall research supervision, project coordination, funding acquisition, and critical revision of the manuscript. All authors read and approved the final version of the manuscript.

Competing Interests

The authors declare that they have no competing interests.

Availability of Data and Materials

The source code and experimental data supporting the results reported in this manuscript are publicly available in the following GitHub repository: https://github.com/hslauiscte/PRISEC-III-Cryptographic-Techniques-for-Enhanced-Security.

References

Aberna, P. and Agilandeeswari, L. (2025). Powbwm: Proof of work consensus cryptographic blockchain-based adaptive watermarking system. *Alexandria Engineering Journal*, 112:510–537. DOI: 10.1016/j.aej.2024.10.016.

Ahn, J., Hussain, R., Kang, K., and Son, J. (2025). Exploring encryption algorithms and network protocols: A comprehensive survey of threats and vulnerabilities. *IEEE Communications Surveys & Tutorials*. DOI: 10.1109/COMST.2025.3526605.

Aliabadi, F., Majidi, M., and Khorashadizadeh, S. (2022). Chaos synchronization using adaptive quantum neural networks and its application in secure communication and cryptography. *Neural Computing and Applications*, 34:6521–6533. DOI: 10.1007/s00521-021-06768-z.

Amrita, Ekwueme, C. P., Adam, I. H., and Dwivedi, A. (2024). Lightweight cryptography for internet of things: A review. *EAI Endorsed Transactions on Internet of Things*, 10. DOI: 10.4108/eetiot.5565.

Azar, K. Z., Kamali, H. M., Homayoun, H., and Sasan, A. (2021). From cryptography to logic locking: A survey on the architecture evolution of secure scan chains. *IEEE Access*, 9:73133–73151. DOI: 10.1109/ACCESS.2021.3080257.

Bhagat, V., Kumar, S., Gupta, S. K., and Chaube, M. K. (2023). Lightweight cryptographic algorithms based on different model architectures: A systematic review and futuristic applications. *Concurrency and Computation: Practice and Experience*, 35(10):e7425. DOI: 10.1002/cpe.7425.

Blackwood, A., Carrington, J., Baryshevsky, S., *et al*. (2024). The implementation of a hybrid large language model for adaptive cryptographic cyber defense. DOI: 10.21203/rs.3.rs-5120507/v1.

Brennaf, M. S., Yang, P., and Lanfranchi, V. (2025). Secured cost-effective anonymous federated learning with proxied privacy enhancement for personal devices. *IEEE Internet of Things Journal*, 12(15):29343–29353. DOI: 10.1109/JIOT.2025.3569200.

Costa, P. and Leithardt, V. (2024). Prisec ii – a comprehensive model for iot security: Cryptographic algorithms and cloud integration. Available at:`https://arxiv.org/abs/2407.16395`.

de Juan-Iglesias, P., Gómez-Gómez, I., Barquero-Jimenez, C., Wilson, C. A., and Motrico, E. (2024). Effectiveness of online psychological interventions to prevent perinatal depression in fathers and non-birthing partners: A systematic review and meta-analysis of randomized controlled trials. *Internet Interventions*, 37. DOI: 10.1016/j.invent.2024.100759.

Dutta, I. K., Ghosh, B., and Bayoumi, M. A. (2019). Lightweight cryptography for internet of insecure things: A survey. In *IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 475–481. DOI: 10.1109/CCWC.2019.8666557.

Farooq, A., Tariq, S., Amin, A., *et al*. (2024). Towards the design of new cryptographic algorithm and performance evaluation measures. *Multimedia Tools and Applications*, 83:9709–9759. DOI: 10.1007/s11042-023-15673-7.

Goyal, R., Pawar, A., Ravikumar, R., *et al*. (2024). A novel hybrid communication policy using network coding. *Wireless Personal Communications*. DOI: 10.1007/s11277-023-10854-x.

Hanchate, R. and Anandan, R. (2024). Medical image en-

cryption using hybrid adaptive elliptic curve cryptography. *IETE Journal of Research*, 70(6):5734–5749. DOI: 10.1080/03772063.2023.2268578.

Khan, B. and Hashmi, A. (2025). Comparing crypto and digital cash systems: A cryptographic analysis. DOI: 10.36227/techrxiv.173627402.26411980/v1.

Kong, J. H., Ang, L.-M., and Seng, K. P. (2015). A comprehensive survey of modern symmetric cryptographic solutions for resource constrained environments. *Journal of Network and Computer Applications*, 49:15–50. DOI: 10.1016/j.jnca.2014.09.006.

Rivadeneira, J. E., Borges, G. A., Rodrigues, A., Boavida, F., and Silva, J. S. (2024). A unified privacy preserving model with ai at the edge for human-in-the-loop cyber-physical systems. *Internet of Things*, 25:101034. DOI: 10.1016/j.iot.2023.101034.

Rivadeneira, J. E., Silva, J. S., Colomo-Palacios, R., Rodrigues, A., and Boavida, F. (2023). User-centric privacy preserving models for a new era of the internet of things. *Journal of Network and Computer Applications*, 217. DOI: 10.1016/j.jnca.2023.103695.

Rubin, I. (2025). Networking security. In *Principles of Data Transfer Through Communications Networks, the Internet, and Autonomous Mobiles*, pages 671–684. IEEE. DOI: 10.1002/9781394267781.ch22.

Sanchez, O. T., Raposo, D., Rodrigues, A., Boavida, F., and Silva, J. S. (2023). Private lorawan network gateways: Assessment and monitoring in the context of iiot-based management. *Engineering Proceedings*, 47(1):4. DOI: 10.3390/engproc2023047004.

Saraiva, D. A. F., Leithardt, V. R. Q., de Paula, D., Mendes, A. S., González, G. V., and Crocker, P. (2019). Prisec: Comparison of symmetric key algorithms for iot devices. *Sensors*, 19(19):4312. DOI: 10.3390/s19194312.

Seok, B. and Lee, C. (2025). A novel approach to construct a good dataset for differential-neural cryptanalysis. *IEEE Transactions on Dependable and Secure Computing*, 22(1):246–262. DOI: 10.1109/TDSC.2024.3387662.

Shanks, G., Sterling, M., Harrington, N., *et al*. (2024). Innovative framework for ransomware detection using adaptive cryptographic behavior analysis. DOI: 10.36227/techrxiv.173214000.03491354/v1.

Silva, C., Cunha, V., Barraca, J., *et al*. (2024). Analysis of the cryptographic algorithms in iot communications. *Information Systems Frontiers*, 26:1243–1260. DOI: 10.1007/s10796-023-10383-9.

Thakor, V. A., Razzaque, M. A., and Khandaker, M. R. A. (2021). Lightweight cryptography algorithms for resource-constrained iot devices: A review, comparison and research opportunities. *IEEE Access*, 9:28177–28193. DOI: 10.1109/ACCESS.2021.3052867.

Wang, Z., Zhang, Z., Wang, Y., and Sun, R. (2025). An integrated evaluation framework of covert performance for lpi signals in iot security. *IEEE Internet of Things Journal*, 12(15):29860–29872. DOI: 10.1109/JIOT.2025.3569527.

Xu, B., Liu, Z., Zhu, H., Dong, B., Zhao, B., Yan, B., and Wei, J. (2025). A novel adversarial attack method for time-series regression models in iiot-based digital twins. *IEEE Internet of Things Journal*, 12(15):29278–29290. DOI: 10.1109/JIOT.2025.3569857.

Yuan, K., Huang, Y., Du, Z., *et al*. (2024). A multilayer composite identification scheme of cryptographic algorithm based on hybrid random forest and logistic regression model. *Complex & Intelligent Systems*, 10:1131–1147. DOI: 10.1007/s40747-023-01212-2.

Zhang, C., Liang, Y., Tavares, A., Wang, L., Gomes, T., and Pinto, S. (2024). An improved public key cryptographic algorithm based on chebyshev polynomials and rsa. *Symmetry*, 16(3):263. DOI: 10.3390/sym16030263.