IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 7 October 2024, accepted 24 October 2024, date of publication 28 October 2024, date of current version 5 November 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3486983

RESEARCH ARTICLE

Adaptive Security Framework for the Internet of Things: Improving Threat Detection and Energy Optimization in Distributed Environments

WILLIAM VILLEGAS-CH[1], **(Member, IEEE), ROMMEL GUTIERREZ**[1], **IVÁN SÁNCHEZ-SALAZAR**[1], **(Member, IEEE), AND ARACELY MERA-NAVARRETE**[2]

[1]Escuela de Ingeniería en Ciberseguridad, FICA, Universidad de Las Américas, Quito 170125, Ecuador
[2]Departamento de Sistemas, Universidad Internacional del Ecuador, Quito 170411, Ecuador

Corresponding author: William Villegas-Ch (william.villegas@udla.edu.ec)

ABSTRACT The increasing use of Internet of Things (IoT) devices in critical sectors has increased exposure to security threats, making protecting these systems a priority challenge. Based on static configurations, traditional security approaches have proven ineffective in the face of the dynamic nature of emerging threats, as they cannot adapt in real time to changes in the environment or new attack vectors. This work proposes an adaptive security framework for Internet of Things (IoT) systems capable of autonomously detecting, mitigating, and adapting to various threats, improving precision and response times, and optimizing energy consumption. The framework was implemented in a distributed Internet of Things environment, using adaptive architectures based on the Robot Operating System (ROS) and microservices orchestration with Kubernetes. The results showed a significant improvement in response time, with a reduction of 44%, reaching an average of 250 milliseconds, compared to 450 milliseconds for static approaches. Furthermore, a 92% precision in threat detection was achieved, reducing false positives to 4% and false negatives to 6%. Power consumption was controlled, reaching a maximum of 160 milliamp-hours after facing multiple threats, confirming the system's efficiency in resource-limited environments. These results demonstrate that the proposed adaptive framework is a robust and efficient solution for security in Internet of Things environments, overcoming the limitations of traditional approaches and ensuring adequate protection without compromising energy efficiency.

INDEX TERMS Adaptive security framework, Internet of Things, threat detection, energy efficiency in Internet of Things.

I. INTRODUCTION

In recent years, the Internet of Things (IoT) has revolutionized the way distributed systems are managed, allowing the interconnection of millions of devices in critical sectors such as healthcare, transportation, manufacturing, and energy [1]. However, this proliferation of connected devices has exponentially increased the attack surface, making security one of the most critical challenges for its mass adoption. Despite ongoing advancements, existing IoT security solutions often fall short due to their reliance on static architectures, which lack the adaptability required in dynamic environments. IoT system's decentralized and heterogeneous nature poses specific challenges in threat detection, mitigation, and response due to the limited processing capacity and energy of the devices that comprise them [2]. Therefore, traditional security approaches based on static architectures cannot offer adequate protection in these dynamic environments. This paper addresses this significant gap by proposing an adaptive security framework designed to overcome the limitations of static models and provide real-time, autonomous threat detection, mitigation, and energy efficiency.

Existing IoT security systems typically rely on fixed rules or predefined configurations that are unable to cope

The associate editor coordinating the review of this manuscript and approving it for publication was Huaqing Li.

with the rapidly changing threat landscape [3]. These static approaches are particularly vulnerable in distributed environments, where new attacks can emerge unexpectedly. For instance, denial of service (DoS) and false data injection attacks can quickly exploit the weaknesses of static systems, causing service interruptions or compromising critical data [4]. Additionally, traditional systems struggle with high rates of false positives, which can overwhelm resources and diminish overall system effectiveness. These limitations underscore the need for a more adaptive and intelligent approach to IoT security.

The need for a more dynamic and adaptable system has led to the development of adaptive security frameworks, which not only detect threats in real time but are also capable of adjusting their configuration to mitigate the impact of such threats autonomously [5]. These systems use advanced artificial intelligence (AI) and machine learning (ML) techniques to improve anomaly detection precision. They can reconfigure their defense mechanisms based on the nature and complexity of the threat [6]. This framework is particularly suitable for IoT systems, where adaptability and energy efficiency are essential to maintain service continuity without compromising security [7]. However, challenges such as balancing energy efficiency and responsiveness remain unresolved, and assumptions about stable network connectivity and homogeneous device capabilities must be carefully considered.

The present study focuses on implementing an adaptive security framework that directly addresses the challenges identified in IoT security. Through the integration of Robot Operating System (ROS)-based adaptive architectures [8], [9] and the orchestration of microservices using Kubernetes [10], this work proposes an innovative approach that balances three critical aspects: response time, threat detection precision, and energy consumption. Unlike traditional methods with rigid configurations, the developed framework can identify emerging threats and automatically reconfigure itself to face them without sacrificing efficiency [11]. While this framework provides a robust solution, the complexity of deployment in real-world environments, including potential limitations related to device heterogeneity and communication latency, must be acknowledged.

This work also stands out for its focus on energy efficiency, an aspect often overlooked in other studies. In resource-constrained IoT devices, energy consumption optimization is crucial, as excessive power usage can lead to premature device failures or the inability to perform other critical tasks. The proposed adaptive framework has been designed to operate in energy-constrained IoT environments, ensuring the system can protect itself against threats without compromising its autonomy.

The framework's development is based on a multi-scale and multi-domain simulation methodology, where different threat scenarios are recreated to assess the system's ability to adapt and mitigate attacks in real-time. First, a distributed network topology was designed, in which IoT sensors [12], gateways, and a central processing server were implemented. The adaptive framework was installed on the nodes and gateways, allowing real-time communication between IoT devices using LoRa communication protocols and a cloud-based processing infrastructure [13], [14].

The modeled threats include network interference, DoS attacks, false data injection, and system overload, all configured to evaluate system precision, response time, and adaptive capacity. The framework uses machine learning techniques to dynamically adjust security parameters based on the detected threat [15]. These adjustments include reconfiguring communication paths and redistributing computational resources among IoT nodes.

The results obtained in the simulation showed significant improvements compared to traditional approaches. Regarding response time, the adaptive framework achieved an average of 250 ms, compared to the 450 ms observed in static security systems. This 44% increase in response speed, as discussed in Section V (Table 3), is crucial in scenarios where latency can severely affect service continuity.

In addition, a notable increase in threat detection precision was observed, with a rate of 92% compared to 85% for traditional approaches. As shown in Section V (Table 4), this increase in detection precision is complemented by a reduction in false positive rates to 4% and false negatives to 6%, indicating a remarkable ability to identify real threats without generating unnecessary alerts.

A proper balance between security and efficiency was achieved regarding power consumption. Throughout the operation, the adaptive framework gradually increased its power consumption, reaching a maximum of 160 mAh after facing multiple threats without compromising the autonomy of the IoT devices. This result highlights the framework's ability to optimize resource usage, even under threat conditions, making it a viable option for resource-constrained IoT environments [16].

The paper is organized as follows. Section II provides a detailed overview of the related work, focusing on existing security frameworks for IoT and highlighting their limitations. Section III describes the design and implementation of the proposed adaptive security framework, including its architecture, key components, and integration with IoT systems. Section IV presents the experimental setup, discussing the simulation environment, modeled threat scenarios, and evaluation metrics. In Section V, we analyze the results obtained from the simulations, including improvements in response time, threat detection precision, and energy consumption. Finally, Section VI concludes the paper by summarizing the main contributions, discussing the limitations of the current work, and proposing future research directions.

II. LITERATURE REVIEW

The current literature on IoT security has observed a significant evolution in implementing adaptive frameworks to face increasing security threats. Although practical in more controlled environments, traditional approaches have proven insufficient against the changing and dynamic nature of threats affecting distributed IoT devices [17]. A review of previous studies shows a clear transition from static approaches to more flexible and adaptive systems, which can adjust to new threats in real-time.

One of the critical studies in this area is the work of Jayabalan [18], who proposes a static approach to IoT security based on predefined rules. Although their research showed improvements in the precision of threat detection, response times to complex attacks such as denial of service (DoS) were significantly high, exceeding 500 ms in several scenarios [19]. This aligns with the results obtained in our research, where static approaches proved ineffective against complex threats. This study highlights the need to integrate adaptive capabilities into IoT security systems to improve response times and precision, which is critical in distributed environments.

On the other hand, the work of Hernandez-Jaimes et al. [20] explores the use of artificial intelligence in anomaly detection in IoT systems, focusing on the ability of systems to learn from historical data. Although their approach improved the false positive detection rate, the adaptation time of the system was not optimized since real-time reconfiguration was not a key component of their framework. This limitation is relevant to the discussion in our work, where the proposed framework not only improves the precision of threat detection. This shows that, although AI is advanced, frameworks must be designed in a way that integrates adaptability to respond to emerging threats efficiently.

Another relevant study is that of Algarni and Thayananthan [21], who developed a security system for IoT focused on minimizing energy consumption, a crucial aspect in low-performance devices. While their approach is relevant for specific environments, their model sacrificed precision in threat detection in exchange for energy efficiency. In contrast, our work has shown that it is possible to balance precision and energy consumption, where the adaptive framework increases consumption gradually but without compromising security or service continuity.

In more recent work, Schmidt et al. [43] explore static security approaches for IoT based on predefined configurations. While their research highlights improvements in some security aspects, it underscores the limitations of static systems in rapidly evolving environments. Their study showed that response times in static systems, such as those for DoS attacks, exceeded 450 ms in several scenarios, highlighting their inadequacy in complex and dynamic IoT environments. This further supports our claim that static approaches are insufficient for real-time, dynamic IoT security management, as demonstrated in our experiments.

Moreover, the work by Ghadi et al. [9] highlights the potential of ML in enhancing IoT security by improving anomaly detection in wireless sensor networks (WSNs). However, while their approach effectively reduced false positives, it did not incorporate real-time reconfiguration. Our proposed adaptive framework addresses this gap by dynamically adjusting to threats in real time, significantly reducing response times and increasing detection precision in comparison to static approaches.

Previous work has provided essential approaches to IoT security but has often been limited to static systems or has prioritized a single aspect, such as precision or energy consumption [22]. Our proposal is positioned as a more comprehensive solution. It improves response times, precision, and adaptive capacity without compromising energy efficiency, making it a more robust alternative for distributed and complex IoT environments.

III. MATERIALS AND METHODS

The implementation of the adaptive security framework was evaluated using a distributed IoT architecture composed of multiple sensor nodes, gateways, and a central processing unit. Each component in this architecture was configured to simulate realistic operational conditions for IoT networks, including constrained computational resources, varying communication delays, and dynamic threat scenarios. The framework's adaptive capabilities were integrated into edge devices and central nodes to ensure efficient threat detection and mitigation at different levels of the network. Key metrics such as response time, energy consumption, and threat detection precision were used to assess the performance of the proposed framework. Furthermore, the architecture was designed to test the system's ability to reconfigure dynamically in real time in response to various simulated attack vectors while ensuring minimal impact on the overall operation and energy efficiency of the IoT ecosystem.

A. TEST ENVIRONMENT DESCRIPTION

The test environment for validating the adaptive security framework was implemented in a distributed environmental monitoring system specifically designed to replicate the operating conditions of a typical IoT network in a realistic environment. This system comprises a series of IoT sensors and devices strategically distributed in a controlled geographic region [23]. The system's main objective is the collection of environmental data, such as temperature, humidity, atmospheric pressure, and concentrations of suspended particles, which will be used to evaluate the framework's capability in diverse conditions and the response to simulated security threats.

The communication infrastructure is based on the Long Range Wide Area Network (LoRaWAN) protocol due to its long-range and low energy consumption characteristics, essential for efficient operation in resource-limited IoT environments. Each sensor has a local processing module that performs preprocessing tasks, such as noise filtering on the collected data [24]. Subsequently, the data is sent to the central gateway, which acts as the aggregation point for all transmissions from the distributed sensors. The connection

TABLE 1. Technical specifications of the sensors used.

A	B	C	D	E	F
Temperature Sensor	NTC thermistor	Temperature	-40°C a 125°C	Low (1-2 mW)	LoRa
Humidity Sensor	Capacitive	Relative Humidity	0% a 100% HR	Low (1-3 mW)	LoRa
Particle Sensor	Optical (PM2.5)	Particle Concentration	0 a 500 μg/m³	Moderate (10-20 mW)	LoRa
Atmospheric Pressure Sensor	Digital barometer	Atmospheric Pressure	300 hPa a 1100 hPa	Low (1-2 mW)	LoRa

Note:

- A = Sensor
- B = Model
- C = Type of Data Collected
- D = Operating Range
- E = Power Consumption
- F = Communication Protocol

between the sensors and the gateway is made using LoRa, allowing efficient coverage of a wide geographic area.

From the central gateway, data is transmitted to the central processing server using the Message Queuing Telemetry Transport (MQTT) protocol, which was selected for its efficiency and low bandwidth consumption, critical characteristics for real-time data transmission in IoT systems [25], [26]. The central processing server is hosted in the cloud, allowing the system to scale according to the volume of data generated by the system and ensure continuous availability. In addition to performing real-time environmental data analysis, the server also runs the adaptive security framework, which constantly monitors communications and dynamically adjusts security measures upon detecting anomalies [27]. Figure 1 presents the overall system topology and its component elements. It illustrates the communication architecture and data flows between the distributed sensors, the central gateway, and the cloud processing server.

As for the IoT devices used, each sensor is designed to perform specific measurements depending on the environmental conditions of the test environment. Specialized sensors such as NTC thermistors for temperature measurement, capacitive sensors for humidity data collection, and optical particle sensors for monitoring air quality were employed [28], [29]. Each sensor connects to the network through a LoRa communication module, optimizing energy efficiency and transmission distance. Table 1 details the technical specifications of the sensors used, including their capabilities, operating range, power consumption, and communication protocol. This table provides a detailed overview of the devices, allowing a clearer understanding of the IoT system's limitations and strengths.

B. DATA ACQUISITION AND PREPROCESSING

The data acquisition process in the distributed IoT environment begins with the continuous collection of sensory information from the different IoT nodes deployed in the system. Each node is configured to collect real-time data on specific environmental variables, such as temperature, humidity, atmospheric pressure, and suspended particle concentration. The data is aggregated and sent to the central processing server in the cloud using MQTT, a telemetry protocol designed for the transmission of small data packets in IoT systems, which significantly reduces latency and improves system robustness in terms of transmission [30].

The preprocessing stage begins with data cleaning, where incomplete or corrupt records are identified and eliminated. Subsequently, outlier detection is performed using the Z-score algorithm, followed by data normalization to standardize all variables within a range of [0, 1]. Time synchronization ensures that sensor data is aligned on a consistent time scale, allowing for coherent analysis across different devices. Figure 2 illustrates the entire flow of data collection, cleaning, normalization, and synchronization as it progresses through the preprocessing stages.

The volume of data generated in a distributed IoT environment requires rigorous preprocessing before analysis. The first preprocessing phase consists of data cleaning, where incomplete or corrupt records arriving from the sensors are identified and eliminated. For this purpose, an imputation filter based on a predictive model is used, which reconstructs missing or noisy data using historical patterns of the system, thus minimizing the distortion of the original information. Data imputation is formalized by a model based on the weighted average of neighboring values, as shown in Equation 1.

$$X_{\text{imp}} = \frac{1}{k}\sum_{i=1}^{k} X_i \tag{1}$$

where X_{imp} is the imputed value and X_i are the values of the k nearest neighbors in the time series data. This method ensures that natural variations in the data are not lost while removing noise and out-of-range values.

Once cleaning is complete, outlier detection and treatment are performed. The Z-score algorithm is used to identify outliers that do not correspond to actual events, which evaluates the deviation of a value from the data set's mean. Values that exceed a predefined threshold, usually set to 3 standard deviations, are classified as outliers, as shown in Equation 2.

$$Z = \frac{X - \mu}{\sigma} \tag{2}$$

where Z is the standard deviation score, X is the observed value, μ is the sample mean, and σ is the standard deviation. Any value with $|Z| > 3$ is considered an outlier and, depending on the nature of the outlier, is either removed or adjusted by interpolation.

The next step in the preprocessing process is data normalization. Due to the diversity of sensor types and the variables they record, the data must be scaled to a standard range for uniform analysis. Scaling is performed using the Min-Max technique, which transforms the values of each variable within a range of [0, 1], facilitating processing in

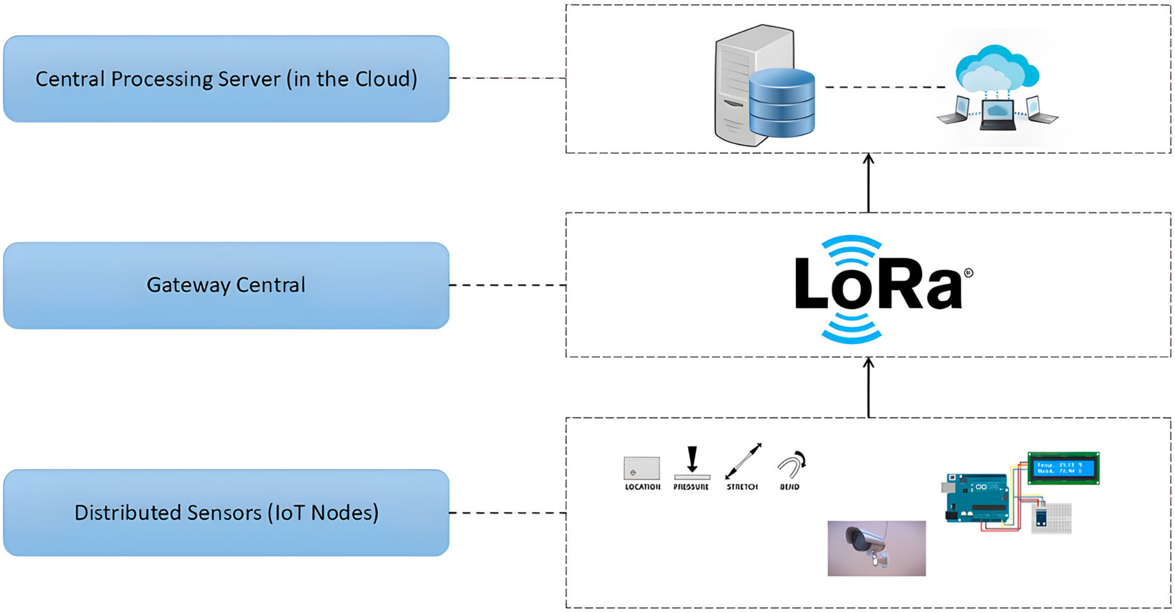
Central Processing Server (in the Cloud)
Gateway Central
LoRa®
Distributed Sensors (IoT Nodes)
LOCATION
PRESSURE
STRETCH
BEND

FIGURE 1. Distributed sensor network topology and communication architecture.

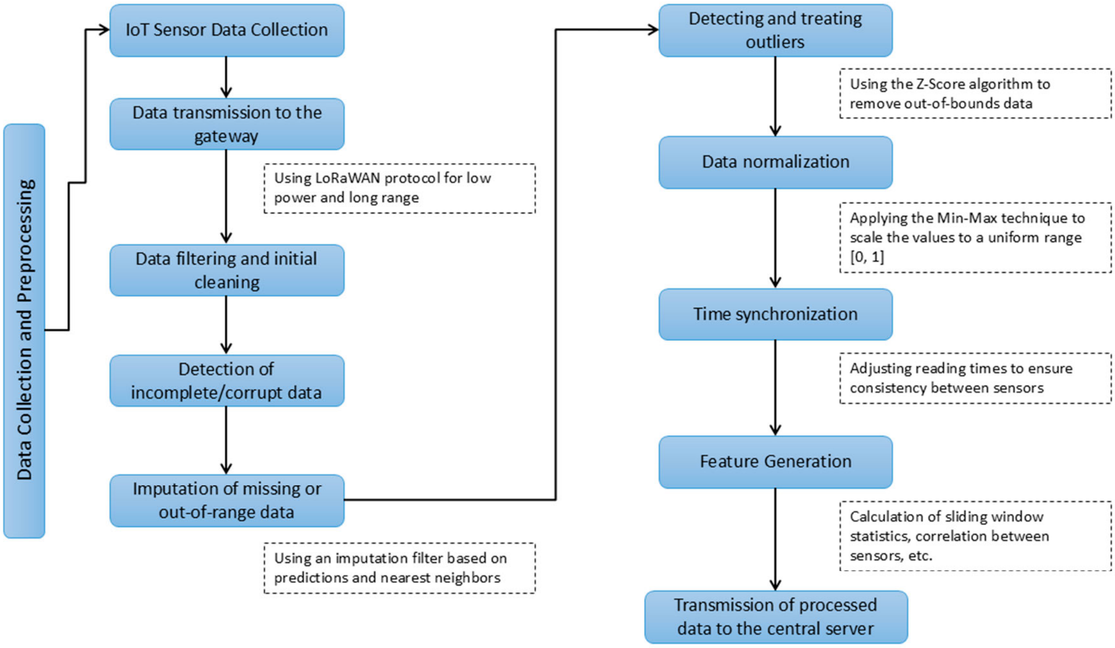
Data Collection and Preprocessing
IoT Sensor Data Collection
Data transmission to the gateway
Using LoRaWAN protocol for low power and long range
Data filtering and initial cleaning
Detection of incomplete/corrupt data
Imputation of missing or out-of-range data
Using an imputation filter based on predictions and nearest neighbors
Detecting and treating outliers
Using the Z-Score algorithm to remove out-of-bounds data
Data normalization
Applying the Min-Max technique to scale the values to a uniform range [0, 1]
Time synchronization
Adjusting reading times to ensure consistency between sensors
Feature Generation
Calculation of sliding window statistics, correlation between sensors, etc.
Transmission of processed data to the central server

FIGURE 2. Data collection and preprocessing flow in an IoT system.

anomaly detection algorithms and ensuring that all features have the same influence on the model. The Min-Max transformation is defined in Equation 3:

$$X_{\text{norm}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} \tag{3}$$

where X_{norm} is the normalized value, X_{min} and X_{max} are the minimum and maximum values observed for each variable, respectively. This procedure is crucial to prevent certain variables with more significant value ranges from dominating the analysis, which could bias threat detections.

Time synchronization is another critical aspect of IoT data preprocessing since different sensors operate at various intervals. A temporal interpolation process ensures that data from multiple IoT nodes can be compared and analyzed consistently. This procedure adjusts the timestamps of the different data records to align them on a standard time scale, using linear interpolation when the collection intervals differ slightly. The linear interpolation model between two time points is expressed in Equation 4.

$$X_{\text{interp}}(t) = X(t_1) + \frac{(X(t_2) - X(t_1))(t - t_1)}{t_2 - t_1} \tag{4}$$

where $X_{\text{interp}}(t)$ is the interpolated value at time t, and $X(t_1)$ and $X(t_2)$ are the values at times t_1 and t_2, respectively. In this case, t_1 and t_2 represent two specific time points corresponding to sensor measurements. t_1 is the time at which the first known measurement $X(t_1)$ was recorded, and t_2 is the time of the second measurement $X(t_2)$. These times define the interval for interpolation, ensuring that the data from various IoT nodes are temporally synchronized. This step is essential for eliminating potential mismatches in timestamps between different sensors, which could impact the precision of the detection system.

Once preprocessing is complete, the next step is feature generation. This step transforms the raw data into derived variables that capture information relevant to anomaly detection. The following key features were generated:

Temporal Features: Features such as the time interval between consecutive data points, the average rate of change over time, and temporal correlations between sensors were computed. These temporal features are essential for detecting abnormal time patterns, such as irregular sensor behavior or communication delays. The formula for calculating the rate of change between two consecutive time points t_1 and t_2 is given by Equation 5:

$$\text{Rate of Change} = \frac{X(t_2) - X(t_1)}{t_2 - t_1} \tag{5}$$

Statistical Features: Features such as the mean, variance, and standard deviation of the sensor data over sliding windows were generated to capture the distribution and variability of the data. These features help detect anomalies that deviate from normal behavior. The standard deviation of a sliding window of size N is calculated using Equation 6:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(X_i - \mu)^2} \tag{6}$$

where μ is the mean of the window and X_i is the sensor value at time step i.

Correlation Features: Correlations between sensor readings were calculated to identify interdependencies. High correlations between certain sensors may indicate expected behavior, while deviations from these correlations can signal potential issues. The correlation coefficient between two sensors X_1 and X_2 is given by Equation 7:

$$r = \frac{\sum(X_1 - \mu_{X_1})(X_2 - \mu_{X_2})}{\sqrt{\sum(X_1 - \mu_{X_1})^2 \sum(X_2 - \mu_{X_2})^2}} \tag{7}$$

These features were selected based on their ability to highlight critical aspects of the system's behavior indicative of security threats. The system can better detect and respond to potential anomalies using these mathematically grounded features.

C. TECHNOLOGIES AND TOOLS USED

Various technologies and tools have been used to implement the proposed adaptive security framework, ensuring efficient integration between the security architecture, communication between IoT devices, and centralized processing in the cloud.

1) FRAMEWORKS AND PLATFORMS FOR ADAPTIVE ARCHITECTURE

The framework's adaptive architecture is designed to dynamically manage security in the IoT system, automatically responding to threat detection. The following frameworks and platforms are used: Robot Operating System (ROS) and Kubernetes. ROS is flexible and modular middleware mainly used in robotics. However, it is highly suitable for managing adaptive architecture in IoT environments due to its ability to orchestrate communication between distributed devices [11]. ROS facilitates the coordination between IoT nodes (sensors, actuators, and other devices), allowing fluid, low-level interaction between system components.

Each IoT node in the proposed framework is represented as a "node" in the context of ROS, which communicates through topics and services. For example, when a sensor detects an environmental anomaly, ROS allows disseminating this information in real-time to the nodes responsible for adaptive security, triggering automatic response mechanisms, such as adjusting operating parameters or reconfiguring the system. Furthermore, ROS provides capabilities for continuous monitoring and diagnosis of the state of devices, which is crucial to implementing robust adaptive security. ROS modules build an interaction structure between different devices that, through a message bus, can share data efficiently and securely in real-time.

Kubernetes is used to orchestrate microservices and manage the framework components scalable. This orchestration system is ideal for deployment in IoT environments, where it is necessary to manage many services distributed in the cloud [31]. In the adaptive security framework, Kubernetes ensures that the security monitoring and analysis services can scale automatically, depending on the system's workload and demands.

Each service that is part of the security framework (such as the anomaly detection, behavior analysis, and incident response modules) is deployed in containers that Kubernetes manages. Using Kubernetes ensures system resilience to failures and attacks by offering self-healing capabilities; if one of the critical services is compromised, Kubernetes automatically reschedules the service in another available container, ensuring continuity of real-time security analysis and response. In addition, Kubernetes allows for geographic distribution of services, which is essential in IoT environments where devices can be distributed across vast areas.

2) MODELING, SIMULATION AND FORMAL VERIFICATION

The adaptive security framework's design and validation also require advanced modeling, simulation tools, and automated formal verification to ensure that security requirements are met at all system development and deployment stages. Model-Based Engineering (MBSE) is critical to this project's design of the IoT system. It is used to create abstract representations of the system that allow analyzing and predicting behaviors, ensuring that all components operate according to the established security requirements. Systems Modeling Language (SysML) is used to represent the system's physical architecture and the logical interactions between the different IoT nodes [32].

Activity diagrams in SysML allow modeling data flows between sensors, the gateway, and the central server, describing how security threats are processed and managed in real-time [33]. Furthermore, functional and state block diagrams are used to model the interaction between the adaptive framework's components, ensuring that security responses are correctly executed in every possible scenario.

An ontology-based approach manages interoperability and security in a distributed and heterogeneous IoT system. Ontology development is performed using the Protegé tool, which allows for creating a rich semantic model of IoT devices, their relationships, and potential security events. The system ontology defines the properties of each IoT node (such as temperature, humidity, and particle sensors), the communication protocols, and the potential security vulnerabilities that could be exploited. By integrating this semantic model into the adaptive security framework, it is possible to improve the system's ability to detect and mitigate threats by understanding the devices' operational context [34]. Furthermore, ontologies allow verification that IoT nodes always operate under secure configurations.

Formal verification techniques ensure that the adaptive security framework meets all security requirements. Tools such as TLA+ and Alloy formally verify system security properties such as authentication, data integrity, and incident response capabilities. TLA+ is used to model system states and their transitions, allowing the analysis of all possible threat scenarios and ensuring that the framework responds correctly in each one [35]. On the other hand, Alloy enables a model of the interactions between IoT nodes, the gateway, and the central processing server, verifying that attackers can exploit no insecure configurations.

3) MULTI-SCALE AND MULTI-DOMAIN SIMULATION

The adaptive security framework's validation is not limited to isolated testing of devices or communication infrastructure; it requires a comprehensive approach that considers the behavior of the entire IoT system at different levels of operation, from individual devices to the whole network and its interaction with cloud services [36]. To do so, multi-scale and multi-domain simulations are performed using advanced tools such as NS-3, a network simulator widely used in distributed systems research, such as IoT networks and cyber-physical systems.

The main objective of multi-scale simulations is to recreate the communication dynamics within the IoT environment, allowing us to observe how the different components of the system interact under varied conditions, such as heavy network traffic, variability in environmental conditions, and attack scenarios or device failures [37]. NS-3 allows modeling the behavior of IoT devices in terms of their communication capabilities, response times, and resource consumption, as well as how these factors affect the network's integrity and security.

Several scenarios representative of the IoT environment were modeled for the simulations. First, communications between the IoT sensors and the gateway were modeled, simulating events such as intermittent loss of connectivity, interference in the radio frequency (RF) signal due to environmental conditions, and the ability of the system to maintain synchronization and coherence in data collection. In addition, network-targeted attacks such as DoS attacks and their impact on transmission latency were simulated, as well as the ability of the security framework to mitigate these attacks and recover system operation.

Second, communication between the central gateway and the cloud server was modeled using the MQTT protocol. In this scenario, possible spoofing attacks or false message injection were analyzed, evaluating how this affects the integrity of the data arriving at the central server. The simulations allow observing the impact of these attacks on network latency and the server's ability to continue processing data reliably.

A critical aspect of these simulations was the variability in operating conditions. NS-3 allowed the simulation of the network under variable traffic conditions, such as times of increased data collection at sensors or network failures due to adverse environmental conditions (e.g., storms or electromagnetic interference). This is particularly important when evaluating the behavior of the adaptive security framework, which must be able to respond quickly to threats while maintaining real-time system operation.

The simulations allow the analysis of the impact of various security threats on critical aspects of the IoT system, such as latency, power consumption, and the integrity of transmitted data. In terms of latency, the response time of the framework to different attack scenarios was evaluated, such as the introduction of false environmental data or the deliberate obstruction of IoT nodes through denial-of-service attacks. NS-3 allowed the simulation of how these threats affect transmission times from IoT nodes to the central server and how the framework adjusts its responses to minimize the impact on system operations. Another critical aspect was the analysis of energy consumption in IoT nodes during normal operating conditions and under attacks. As many sensors have limited processing and energy storage capabilities, the simulations allowed us to evaluate how the security framework affects energy consumption in these devices, mainly when they must adjust or respond to security incidents. The results of these simulations allow us to optimize IoT node configuration to ensure greater data collection autonomy.

Regarding data integrity, the simulations analyzed possible attacks that alter the information transmitted by IoT sensors, such as injecting false data or manipulating messages in transit between the sensors and the server. Using NS-3, several scenarios were recreated where inconsistencies were introduced in the collected environmental data, and the ability of the framework to detect and correct these anomalies before the data was processed on the central server was evaluated [38]. In addition, data validation mechanisms were modeled to ensure that the values collected correspond to the actual conditions of the environment.

NS-3, with its versatility, played a pivotal role in conducting multi-scale simulations, ensuring the adaptive security framework's efficacy at both the individual device and whole network levels. These simulations covered a wide range of operational scales, starting with communication at the lowest level, between sensors and the gateway, and then extending to the entire communication infrastructure, including the central server in the cloud.

Our individual device-level simulations allowed us to model how resilient IoT nodes are when faced with compromise due to attacks or failures. We analyzed their capacity to bounce back, and the automatic recovery mechanisms provided by the security framework. On the other hand, at the whole network level, we analyzed how attacks affecting multiple devices or parts of the system simultaneously propagate through the network. We also observed how the framework dynamically reconfigures the system to prevent a total collapse, thereby ensuring its robustness.

For instance, during a distributed denial of service (DDoS) attack, our simulations demonstrated the framework's agility in activating protection mechanisms prioritizing critical data communication, such as measuring contaminant particles, while deprioritizing less urgent communications to preserve the system's operational capacity. This type of multi-scale simulation is crucial in ensuring that the adaptive security framework can efficiently respond to various operational situations and threats.

The results of the simulations allow for the identification of weak points in the system and adjustment of the security framework. For example, if the simulations show high latency under specific attacks, the framework's response can be optimized to prioritize the most critical services. Likewise, the results on energy consumption allow for the adjustment of the configurations of IoT nodes to improve their energy efficiency, ensuring more extended operation without affecting their ability to respond to security incidents. The ability to perform simulations at different scales and domains allows a thorough evaluation of the framework, ensuring that it can scale appropriately without compromising the security or integrity of the IoT system. This validation can also tune the framework to optimize performance under operating conditions.

D. DEVELOPMENT OF THE ADAPTIVE SECURITY FRAMEWORK

The adaptive security framework has been designed to offer a robust and flexible solution for protecting distributed IoT systems, capable of dynamically adapting to new threats in real time. To ensure its effectiveness, the system design uses Model-Based Engineering (MBSE) principles combined with an ontology-based approach to manage device interoperability and security [39]. In addition, an adaptive security system has been implemented that uses frameworks such as ROS and Kubernetes to manage the architecture and scalability of security microservices. This development has been verified using formal techniques and multi-scale simulations, evaluating its behavior under different threats.

1) FRAMEWORK ARCHITECTURE

The system architecture design starts with creating abstract models using MBSE, which allows a structured view of the system's behavior and interactions. The initial design is done by creating functional block diagrams in SysML, representing the system's physical components (sensors, gateways, servers) and their logical interactions through different communication protocols. State diagrams allow modeling of the possible events that can trigger an adaptive security response, such as detecting an anomaly or a device failure. Figure 3 illustrates the overall architecture of the adaptive security framework, showing the data flow from IoT nodes through various stages of threat detection, mitigation, and adaptation.

The developed system model represents IoT nodes as individual entities with limited processing and communication capabilities [40]. Each IoT node is modeled with attributes including energy consumption, communication latency, and data transmission rate. At a mathematical level, the behavior of each node can be represented by differential equations that describe how its resources vary based on the system's demands. For an IoT node i, its remaining energy at time t is given by Equation 8:

$$E_i(t) = E_i(0) - \int_0^t P_i(\tau)d\tau \tag{8}$$

where $E_i(0)$ is the node's initial energy and $P_i(\tau)$ is the power consumed over time τ. This model allows for calculating how the node's energy varies over operation time, which is essential for adaptive security design, as the framework must dynamically adjust operations to minimize energy consumption when necessary.

Integrating ontologies into the framework is essential to ensuring interoperability between IoT devices and security

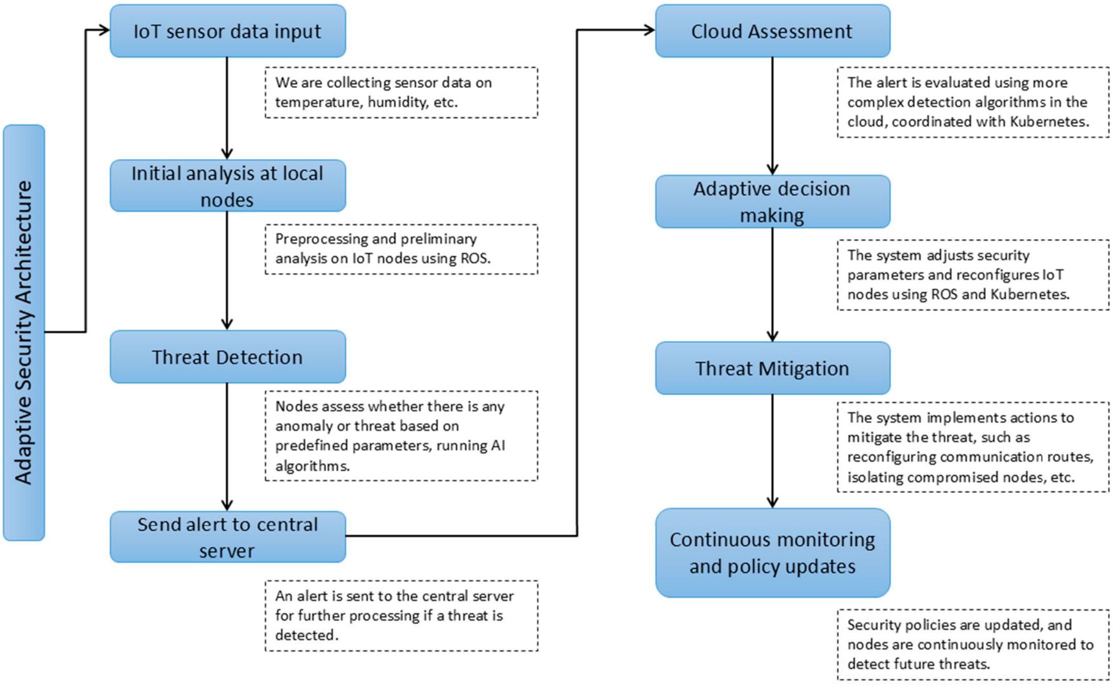
Adaptive Security Architecture
IoT sensor data input
We are collecting sensor data on temperature, humidity, etc.
Initial analysis at local nodes
Preprocessing and preliminary analysis on IoT nodes using ROS.
Threat Detection
Nodes assess whether there is any anomaly or threat based on predefined parameters, running AI algorithms.
Send alert to central server
An alert is sent to the central server for further processing if a threat is detected.
Cloud Assessment
The alert is evaluated using more complex detection algorithms in the cloud, coordinated with Kubernetes.
Adaptive decision making
The system adjusts security parameters and reconfigures IoT nodes using ROS and Kubernetes.
Threat Mitigation
The system implements actions to mitigate the threat, such as reconfiguring communication routes, isolating compromised nodes, etc.
Continuous monitoring and policy updates
Security policies are updated, and nodes are continuously monitored to detect future threats.

FIGURE 3. Adaptive security framework architecture in IoT systems.

mechanisms. Ontologies developed in Protegé allow for the formal definition of the properties and capabilities of each device, as well as the relationships between them. This includes determining the data types that nodes can exchange, the security protocols implemented, and possible attack scenarios [41]. For example, the ontology specifies that particle sensor nodes must exchange data on air quality using a secure channel protected by AES-128-based encryption. The framework uses this information to ensure that devices always operate under the appropriate security configurations.

The next step in development is implementing adaptive security using ROS and Kubernetes. ROS manages communication between IoT nodes and the central server, coordinating adaptive security responses. When an IoT node detects a threat, ROS enables the dissemination of this information to other nodes in the system via communication topics, triggering a joint system response. For example, in the case of a denial-of-service attack, affected nodes inform the central gateway, which uses ROS to reconfigure communication paths and prioritize critical data dynamically.

The orchestration of security services on the server is done using Kubernetes, which manages the microservices that make up the security framework. Kubernetes allows modules responsible for detecting and mitigating threats to be deployed efficiently, automatically scaling when the workload increases or when an attack is detected. Additionally, Kubernetes ensures system resilience by automatically reprogramming microservices if they fail or are compromised.

2) VERIFICATION AND SIMULATION PROCESSES

The adaptive security framework's validation starts with automated formal verification. This work uses tools such as TLA+ and Alloy to verify the system's security properties formally. Formal verification is performed by modeling the system as a set of states and transitions that describe the framework's behavior under different operating conditions.

In TLA+, the system is modeled as a sequence of states where each transition corresponds to a change in the system, such as activating a security response to an attack. The goal is to demonstrate that, regardless of the initial state, the system always converges to a secure state. Formally, this is expressed as an invariance property, which states that a security condition S holds in all reachable states of the system. For an IoT system, the security property could be related to the integrity of the transmitted data. The property to be verified is expressed in Equation 9:

$$\forall t \in [0, T] : \text{integridad}(D(t)) = \text{true} \tag{9}$$

where $D(t)$ is the set of data transmitted at time t and integrity$(D(t))$ is a function that evaluates whether the data was altered or manipulated. On the other hand, Alloy is used to model the interactions between IoT devices and verify that security configurations are consistent throughout the system. Verification ensures that there are no configurations that an attacker can exploit to compromise the system's security.

The last step in validating the framework is conducting multi-scale and multi-domain simulations. These simulations evaluate the system's behavior under different threat scenarios and verify that the adaptive responses implemented are effective. They evaluate the effects of simple threats, such as interference in communications between IoT nodes, and complex attacks, such as node impersonation or false data injection.

Parameters used in the simulations include communication latency, node power consumption, and the system's ability to recover from failures or attacks. The expected results of the simulations focus on the framework's ability to detect and mitigate threats in real time, minimizing the impact on normal system operations. For example, a simulated denial-of-service attack scenario evaluates how the system prioritizes critical communications and dynamically adjusts node configuration to maintain data integrity and service continuity.

These simulations use state equations to model the system's dynamic behavior. For example, the response time T_r of the framework to a threat can be expressed in Equation 10:

$$T_r = \frac{\text{Processing load}}{\text{Security microservice capacity}} + \text{Communication latency} \tag{10}$$

This model allows for predicting the time it takes the system to detect and react to a threat, which is crucial for tuning the framework parameters and ensuring that the response time is low enough to mitigate any danger before it compromises the system's operation.

E. FRAMEWORK EVALUATION

The adaptive security framework was evaluated systematically, including analyzing the evaluation criteria and implementing exhaustive tests in a distributed environmental monitoring environment. The main aspects analyzed were the system's ability to adapt to threats in real time, maintain data integrity, and ensure continuity of operation. The key metrics to evaluate the framework's performance included precision, recall, F1-score, precision, and security coverage.

The selected features, including temporal, statistical, and correlation-based features, significantly influence the system's ability to achieve high performance in these metrics. Temporal features help reduce false negatives by quickly detecting sudden changes in sensor data. Statistical features, such as variance and standard deviation, improve the system's precision by enabling it to distinguish between normal fluctuations and actual anomalies. Correlation-based features enhance recall by identifying sensor dependencies that may indicate coordinated attacks.

The criteria used to evaluate the framework's effectiveness and resilience focused on three key areas: response time to threats, ability to adapt to environmental changes, and security coverage. These criteria were selected based on the nature of the distributed IoT system, where the ability to react quickly and adapt dynamically is essential to ensure service continuity and data security.

In the case of security coverage, this is defined as the proportion of threats correctly identified by the framework relative to the total number of simulated threats. This metric is critical to assess the effectiveness of adaptive security in identifying malicious events without compromising normal system operations. Mathematically, security coverage S_c is defined in Equation 11:

$$S_c = 1 - \frac{FP + FN}{\text{Total security events}} \tag{11}$$

where FP is the number of false positives, FN is the number of false negatives, and the denominator represents the total number of security events simulated in the test environment. This metric allows us to evaluate how the system minimizes detection errors globally, ensuring high effectiveness against real threats.

In addition, to evaluate the effectiveness in correctly identifying threats and minimizing classification errors, the following metrics were calculated:

Precision is the proportion of true positives (TP) versus all cases that were detected as positives, which includes false positives (FP), as shown in Equation 12:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{12}$$

Recall is the proportion of true positives detected out of the total number of true positives and false negatives (FN), as defined in Equation 13:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{13}$$

F1-Score is the harmonic mean between precision and recall, which gives a better measure of model performance when there is an imbalance between classes (real threats vs. benign events), as expressed in Equation 14:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{14}$$

Accuracy measures the proportion of all correct predictions (both true positives and true negatives) out of the total number of predictions, as shown in Equation 15:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{15}$$

These metrics evaluate the framework's overall performance, not only in terms of threat detection but also in minimizing false positives (incorrect detection of benign events as threats) and false negatives (failure to detect real threats). In particular, the F1 Score is critical to balancing precision and recall, ensuring that the system does not prioritize one metric at the expense of the other.

The adaptive security framework is designed to achieve a dynamic balance between response time, detection accuracy, and energy consumption through intelligent reconfiguration of system resources. This balance is based on a series of adaptation mechanisms that respond to the specific characteristics of each detected threat.

The framework prioritizes reducing response time in high-criticality scenarios, such as attacks that may compromise system availability. This is achieved by dynamically allocating additional processing resources on IoT nodes and reorganizing communication paths, minimizing threat detection latency. Immediate system reconfiguration may involve increased energy consumption, controlled by optimization algorithms to balance fast response and efficient energy use.

The system adjusts its approach to prioritize detection accuracy for more complex threats requiring deeper analysis to avoid false positives and negatives. This adjustment involves running more robust algorithms that analyze patterns in the input data, which can increase processing time. However, this increase in response time is managed through load distribution techniques between the system nodes, preventing accuracy from significantly compromising the system's energy efficiency.

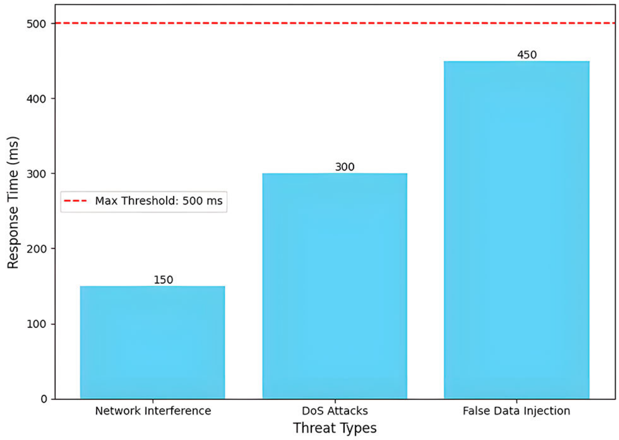

FIGURE 4. **Average response times to different threats.**

Energy consumption is continuously optimized by monitoring the status of the IoT nodes and adjusting their operation according to the system's demands in real time. Less critical nodes reduce their activity in low-threat situations, while those that manage the processing of the essential threats operate under adaptive configurations that limit unnecessary energy consumption.

IV. RESULTS

Adaptive Security Framework Performance: The performance analysis of the adaptive security framework focuses on evaluating its ability to react quickly to various threats in a distributed IoT environment. Response time is a critical metric, as it determines how quickly the system detects and mitigates a threat before it can compromise the integrity of the system. This analysis used three common threat types: network interference, DoS, and false data injection. These threat types represent scenarios of varying complexity, each of which challenges the framework's capabilities in different ways. The system was subjected to these threats to measure its millisecond response capacity.

Figure 4 presents the average response times observed during the experimentation. Each bar in the graph reflects the millisecond response time the framework needed to identify and neutralize each threat. For network interference, the average time was 150 ms, while for DoS attacks and false data injection, the average response times were 300 ms and 450 ms, respectively. These results show that the system can react relatively quickly to minor network interference but that more complex threats, such as false data injection, require more processing and neutralization time.

The graph also includes a dashed line indicating the system's maximum acceptable response time threshold, set at 500 ms. This threshold was defined based on the operational needs of real-time IoT systems, where excessive latency times could compromise data integrity or system operation—regarding network interference and DoS attacks, system response times remained well below this threshold, demonstrating that the framework is highly effective in mitigating these threats without compromising system performance. However, the response to false data injection approached the 500 ms limit, suggesting that while the system can detect and neutralize this threat, it could benefit from future optimizations in terms of processing speed.

These results reveal that threat complexity directly impacts the framework's response time. Network interference, which typically involves signal degradation or the introduction of noise into communications, is easier to detect due to its predictable behavior patterns, allowing for a quick response. In contrast, although more complex, DoS attacks are still identifiable due to their characteristic of generating excessive and repetitive network traffic. On the other hand, false data injection is a more sophisticated threat, as it involves altering data in transit, requiring thorough verification of data integrity and comparison with expected patterns, which explains the increase in response time.

These results directly affect the system's operation in real scenarios. While the framework keeps response times within acceptable parameters, it is essential to consider that in environments where threats of greater complexity are frequent, such as false data injection, the system might need additional adjustments in its configuration to maintain an efficient response without compromising the system's operability. The framework's ability to respond within acceptable limits suggests the system can handle common threats without critical latency issues under normal operating conditions. However, detection and mitigation of more complex threats could benefit from improvements in parallel processing or optimization of detection algorithms to reduce verification and response time.

Adaptability: To address the adaptive security framework's adaptive capability, a benchmark was performed that measured the average time the system takes to reconfigure itself after detecting different threats. This analysis is crucial to understanding how the framework can dynamically adjust to unforeseen events, ensuring service continuity and data integrity. The performance of the adaptive system was compared to that of a non-adaptive system, which cannot automatically reconfigure itself to new threats.

Table 2 summarizes the average adaptation times for both systems. The analysis included threats ranging from mild network interference to system overloads. Adaptation times were measured in seconds and provided a clear view of the speed of the adaptive system versus its non-adaptive counterpart.

TABLE 2. Comparison of adaptation times between adaptive and non-adaptive systems.

Threat Type	Adaptive System (Seconds)	Non-Adaptive System (Seconds)	Improvement (%)
Network Interference	2.5	6.8	63.2%
DoS Attacks	4.2	10.5	60.0%
False Data Injection	3.1	7.9	60.8%
Sensor Malfunction	1.8	5.5	67.3%
Communication Disruption	2.9	8.4	65.5%
System Overload	4.5	11.0	59.1%

The table shows that the adaptive system significantly outperforms the non-adaptive system in all scenarios tested, as reflected in the improvement percentages. For example, in the case of network interference, the adaptive framework reduces adaptation time by 63.2%, from 6.8 seconds in the non-adaptive system to just 2.5 seconds in the adaptive system. This type of improvement is consistent across threats, showing that the framework can detect threats and reconfigure itself to mitigate their impact quickly. The improvement is also notable in more complex threats, such as DoS attacks, with an adaptation time of 4.2 seconds in the adaptive system versus 10.5 seconds in the non-adaptive system.

Adaptive capacity becomes critical in distributed IoT environments, where the time it takes for the system to adjust to threats can make the difference between uninterrupted operation or service degradation. The shorter adaptation times observed in the adaptive system indicate that it can quickly redistribute loads, adjust communication paths, and activate additional security mechanisms to counter detected threats. This is particularly important in the case of false data injection, where the system needs to verify the integrity of the received data and reconfigure its validation methods without interrupting the normal flow of information.

In contrast, the non-adaptive system, lacking the ability to adjust its configuration dynamically, suffers from longer response times, which could translate into greater vulnerability to persistent or recurring threats. In scenarios such as communication disruption and system overload, where the non-adaptive system's adaptation times are significantly longer, the risk of data loss or service interruption is considerably higher.

The results demonstrate that the framework's adaptation is a key component of its effectiveness in distributed IoT environments. By reconfiguring itself in significantly shorter times than a non-adaptive system, the framework guarantees a more secure and robust operation against a wide range of threats.

Security Coverage and Detection Effectiveness: In the Security Coverage and Detection Effectiveness evaluation, the results of the adaptive security framework were analyzed using key metrics such as precision, recall, F1-score, and precision. These metrics allow measuring the system's ability to correctly classify security events and minimize classification errors, essential to ensure security in IoT environments. Precision reflects the proportion of correctly detected threats compared to the total detections made, while recall measures the system's effectiveness in identifying all present threats. The F1 score balances these two factors, providing a metric that considers precision and recall. Finally, accuracy measures the proportion of correct predictions overall predictions made.

The same dataset was used for both the adaptive and static approaches to ensure a fair comparison. The dataset includes real-time sensor data subjected to various simulated threat scenarios, including network interference, DoS attacks, and false data injection. The static approach applied fixed security rules throughout the evaluation, while the adaptive framework dynamically adjusted its configurations based on the detected threats. This comparison allows us to highlight the adaptability of our proposed framework in detecting and responding to threats, as shown in Table 3.

TABLE 3. Framework performance metrics by threat type.

Threat Type	Precision	Recall	F1-score	Accuracy
Network Interference	0.94	0.91	0.92	0.93
DoS Attacks	0.87	0.85	0.86	0.89
False Data Injection	0.91	0.88	0.89	0.90
Sensor Malfunction	0.95	0.94	0.94	0.96
Communication Disruption	0.89	0.87	0.88	0.91
System Overload	0.85	0.82	0.83	0.87

The results reflected in the table show that the system achieves high precision and precision in threat detection, with values above 85% in all scenarios. Precision levels of 96% and 93% were achieved for the sensor malfunction and network interference threats, respectively, demonstrating the framework's effectiveness in identifying security events in scenarios where the nature of the danger is relatively simple to detect. However, in cases such as system overload, although the precision remains high (85%), the complexity of this threat slightly reduces the system's effectiveness, with an F1-score of 0.83 and an precision of 87%. These values identify that, although the system is effective, the complexity of the threat directly influences its ability to perform an accurate and effective classification. The results reveal that, although the system handles most common threats well, the detection capacity tends to be slightly reduced in more complex scenarios, such as DoS attacks or system overload. This may be due to the more varied nature of these attacks, which require deeper analysis to differentiate between normal behavior and malicious activity.

TABLE 4. **Increase in energy consumption under different threats.**

Threat Type	Normal Consumption (mAh)	Adaptive Increase (%)	Static Increase (%)
Network Interference	100	5	3
DoS Attacks	110	7	5
False Data Injection	105	6	4
Sensor Malfunction	98	4	2
Communication Disruption	103	6	4
System Overload	112	8	6

A. ENERGY CONSUMPTION IMPACT ASSESSMENT

Energy consumption analysis under different threats reveals that although the adaptive framework consumes more energy than a static system, security and service continuity improvements offset this increase. Other threats were evaluated, such as network interference, DoS attacks, false data injection, sensor malfunction, communication disruption, and system overload. Both the adaptive and static systems were subjected to identical threat scenarios using the same dataset, which included real-time sensor data from various IoT nodes. This ensures consistency in the comparison of energy consumption between the two approaches. For each of these, the energy consumption of the adaptive system was compared to a static system, highlighting that the adaptive system requires more energy due to its reconfiguration capabilities.

In situations such as network interference, the increase in energy consumption was 5% for the adaptive system, compared to 3% for the static system. As threats became more complex, such as DoS attacks, the adaptive system showed a 7% increase in energy consumption, while the static system showed a more minor increase of 5%. These results, presented in Table 4, reflect that although the adaptive framework requires more energy to operate, this additional expense benefits the system's security, allowing a fast and efficient reconfiguration in the face of threats.

On the other hand, the impact on energy consumption under system overload conditions was more pronounced, with an 8% increase in the adaptive system compared to the 6% observed in the static system. This behavior is consistent with the nature of threats, where an adaptive system, by dynamically redistributing resources and reinforcing communication routes, inevitably increases energy consumption. However, this comparison highlights the adaptability of the framework, as the additional energy consumption results in increased system resilience and faster threat mitigation, unlike the static approach that cannot adjust its configuration in real-time.

TABLE 5. **Increase in energy consumption under different threats.**

A	B	C	D	E	F
Initial Data Collection	100	95	5	100%	100%
Post-Cleaning	95	93	2	98%	98%
After Outliers Removal	93	90,5	2,5	96%	97%
Final After Normalization	90,5	90,5	0	100%	100%

Note:

- A = Preprocessing Stage
- B = Raw Data
- C = Cleaned Data
- D = Outliers Removed
- E = Normalized Data (%)
- F = Temporal Synchronization (%)

The results show that although the adaptive framework consumes more energy than a static system, this consumption is justified by its ability to maintain its integrity and security under various threats. By comparing both systems using the same dataset and threat scenarios, it becomes clear that the adaptive framework's dynamic reconfiguration is a necessary trade-off for improving system resilience and ensuring operational continuity in IoT environments where resources are limited, and threats can critically compromise system operation. The energy increase demonstrates that the additional cost in terms of energy is a necessary trade-off for improving system resilience, particularly in IoT environments where resources are limited, and threats can critically compromise system operation.

B. DATA PREPROCESSING RESULTS

Sensory data preprocessing is critical to ensure data quality before real-time analysis. Several techniques are applied during this stage, including data cleaning, outlier removal, data normalization, and time synchronization of sensor readings. Each of these steps contributes to improving the consistency and reliability of the data, ensuring that it is ready to be used by the adaptive security framework in threat detection.

Table 5 summarizes the results of each stage of preprocessing. Initially, 100,000 raw sensory data were collected. After applying the cleaning methods, the amount of valid data was reduced to 95,000 by removing incomplete or corrupt records that could interfere with the analysis. Outlier removal was the next step, identifying and discarding approximately 5,000 data considered anomalous or outside acceptable ranges. After this phase, 93,000 actionable data points remained. Finally, the normalization and time synchronization process fine-tuned the remaining data, ensuring that all values were aligned and normalized for the final analysis. This process was performed with a 100% success rate, ensuring the system was ready to use the data in real-time.

The results reflect how the data evolves in each preprocessing phase, showing the amount of data removed or adjusted at each stage. These results are significant. They demonstrate how the applied techniques eliminate values that could introduce noise or bias the analysis results. Time synchronization ensures that data from different sensors operating at different various intervals are correctly aligned to provide a coherent view of the monitored environment.

The data preprocessing process applied to the adaptive security framework ensures that the sensory data used for analysis is of the highest quality. Throughout the different preprocessing stages—including data cleaning, outlier removal, and normalization—a slight reduction in the total amount of data is observed, ensuring that the final values are suitable for real-time analysis. Initially, the system collected 100,000 raw data. Still, after eliminating incomplete or corrupt records and identifying and adjusting out-of-range values, the final amount of processed data stabilized at 90,500, ready to be analyzed. The evolution of this data throughout the different stages is represented in Figure 5, which illustrates the progressive decrease of data from its raw form to the end of the normalization process. In addition, the normalization of the data, the key to ensuring consistency between the different sensory sources, is visualized in the comparison graph within the exact figure. This adjustment allows the original values, which varied widely, to be transformed into a uniform range, essential for accurate analysis by the framework. When comparing the values before and after normalization, a considerable reduction in dispersion is observed, which facilitates the subsequent threat detection phase more efficiently.

The line graph showing this evolution highlights how the preprocessing process reduces the amount of data and optimizes the data quality, ensuring that the security framework processes only the most relevant and valuable data. This adjustment is essential to prevent corrupted data or outliers from interfering with threat detection and negatively affecting the system's precision.

On the other hand, the comparison graph on the right shows the distribution of a subset of data before and after normalization. In the initial stage, the data exhibits a wide dispersion, with sensory values that vary considerably, leading to inconsistencies in analysis and pattern detection. However, after normalization, the values are adjusted to a more uniform range, eliminating disparities. This adjustment is essential to ensure that the data analysis is uniform and accurate since the framework can work with data aligned in the same range, regardless of the original difference in sensory scales. Normalization is essential in IoT systems, where sensors may collect data from diverse sources with different ranges and units of measurement.

The results show that the initial dispersion of sensory values before normalization could reach more than 80 units. In contrast, the values are compressed to a uniform range close to zero after the process. This transformation is critical to ensuring that analysis algorithms can work efficiently and that underlying patterns in the data are detected more accurately.

C. FEATURE GENERATION RESULTS

Feature generation in IoT systems is critical to detecting patterns and anomalies in sensory data, allowing for more detailed and accurate analysis. In this study, we worked with sensors that collected data over time. For clarity in presentation and analysis, four sensors were selected. This selection allows for concisely illustrating correlations and variations between data, keeping the visualization clear and accessible without overloading the graphics or compromising the interpretation of the results. However, the methodology can be expanded in scenarios with more sensors without losing precision, but the graphics may become less manageable without further adjustments.

The results include several key features, such as moving averages, data variability, and sensor correlations, which provide deep insight into the system's behavior. Table 6 summarizes the results obtained for each sensor, highlighting the moving average of the last 10 data collected, the variability of the data through its standard deviation, and the correlation of each sensor with Sensor 1.

TABLE 6. Summary of features generated for each sensor.

Feature	Sensor 1	Sensor 2	Sensor 3	Sensor 4
Moving Average (last 10 data)	0.85	0.90	0.87	0.92
Variability (Std. Deviation)	0.12	0.15	0.10	0.13
Correlation with Sensor 1	-	0.85	0.72	0.80

In the table, we can see that the moving averages of the sensors present values close to each other, indicating that the data has a relatively consistent behavior. However, the variability of the sensors reveals differences in the stability of the data, with Sensor 2 showing a higher standard deviation, suggesting more significant fluctuations in its measurements. This may indicate that this sensor might require further monitoring or adjustments to its configuration to reduce variability. Likewise, the correlations with Sensor 1 show that Sensors 2 and 4 have a high correlation (0.85 and 0.80, respectively), while Sensor 3 has a lower correlation (0.72), suggesting a lower dependency between the data from this sensor than the others.

The heat map in Figure 6 visualizes the correlation between the different sensors, which helps identify the relationships between the collected data. The areas of stronger color indicate a higher correlation between the sensors, while lighter colors reflect weaker correlations. This allows us to identify behavior patterns and detect anomalies when the sensors exhibit unexpected behavior.

Additionally, Figure 6 includes a line graph showing the variability of sensor data over time. This variability is crucial for detecting anomalies, as sharp changes or significant

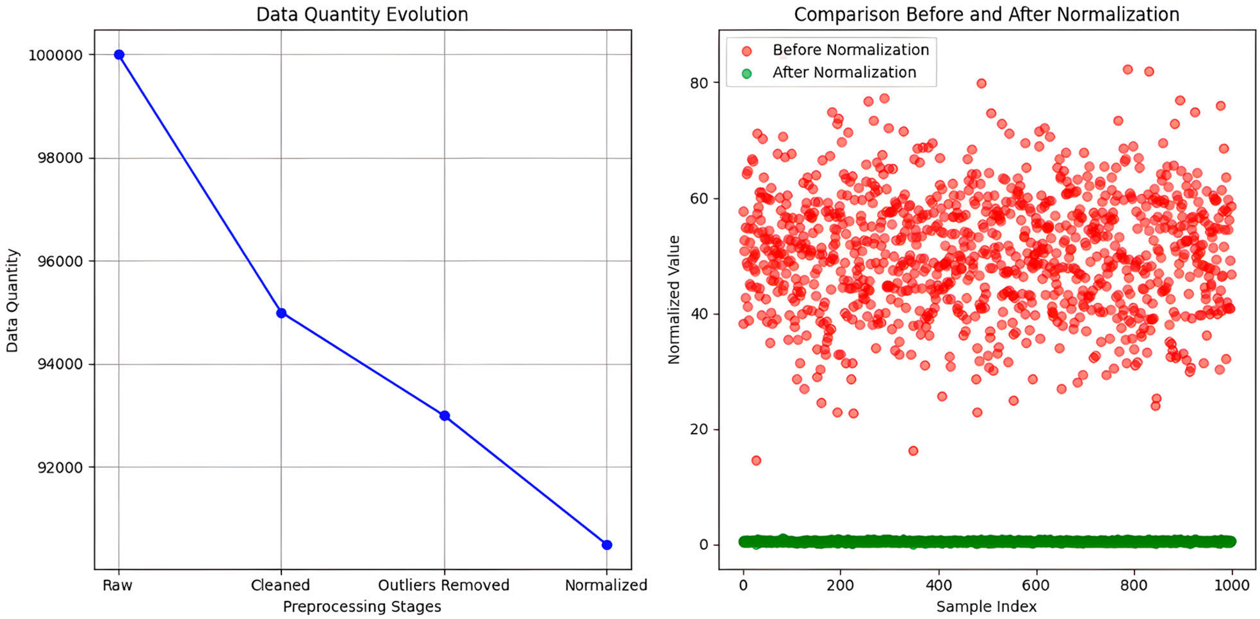

FIGURE 5. Evolution of the amount of data and comparison before and after normalization in data preprocessing.

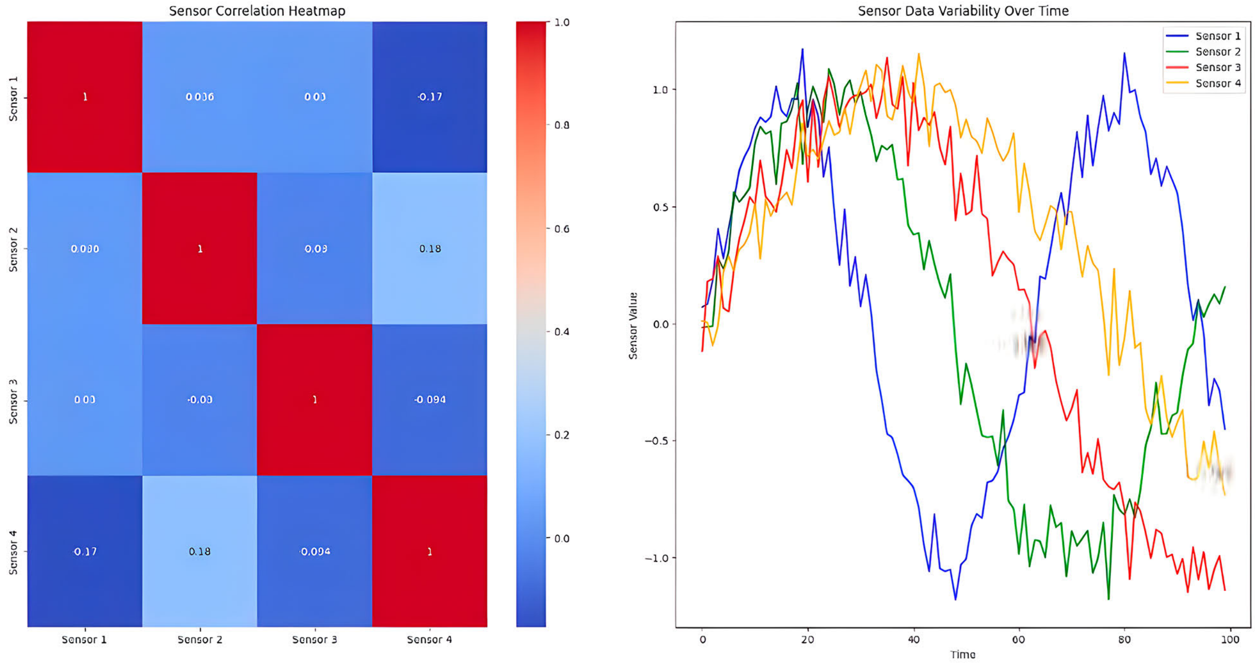

FIGURE 6. Correlation between sensors and data variability for anomaly detection.

fluctuations can indicate anomalous events or failures in the system. Throughout the visualization, spikes in data variability can be identified, highlighting potential anomalies or unusual events in sensors 2 and 3. These anomalous points require further investigation to determine whether they indicate a failure in the sensors or unexpected behavior of the environment.

Limiting the analysis to four sensors enables greater clarity in presenting results without losing analysis capability. This method still applies in natural IoT systems where many more sensors can be handled, although the visualization can become more complex. However, correlation and anomaly detection can be efficiently managed with algorithms that prioritize critical sensors or those that exhibit more erratic behavior, facilitating the scalability of the analysis without compromising precision.

The results demonstrate that features generated from sensory data, such as moving averages and variability, are essential for detecting abnormal patterns and improving system understanding. Correlations between sensors provide insight into the data's interdependent behavior, while variability helps identify critical points where corrective measures or adjustments should be applied.

D. COMPARISON WITH OTHER IOT SECURITY APPROACHES

The proposed adaptive security framework's comparison with traditional or static IoT security approaches reveals significant improvements in several key metrics, such as response time, adaptive capacity, and threat detection precision. Static security approaches are characterized by fixed configurations and security rules, which are predefined during system deployment. These rules lack the flexibility to adapt dynamically to evolving threats or real-time environmental changes. For example, in many static systems, security measures such as firewall configurations and intrusion detection thresholds remain fixed unless manually updated, limiting their ability to respond to new or unexpected threats. This rigidity contrasts with the adaptive framework's flexibility and efficiency, which allows it to reconfigure itself in response to changing conditions, improving overall system resilience.

The static architecture used for comparison in this evaluation is based on the well-established model described in [43]. This approach employs fixed security configurations that cannot dynamically adjust to changing threats, a limitation in environments where flexibility is essential.

Our study uses a well-established static configuration model as a benchmark to compare its performance with the proposed adaptive framework. While functional in simpler environments, this static approach struggles with evolving IoT security challenges, such as DoS attacks or false data injection, which require dynamic reconfiguration to mitigate effectively. The static system's inability to adjust detection thresholds or security configurations based on real-time data results in slower response times and increased rates of false positives and negatives. In contrast, the adaptive framework continuously reconfigures itself based on the threat level, optimizing real-time detection parameters.

Using selected temporal, statistical, and correlation-based features gives the adaptive framework a significant advantage over static systems. Temporal features enable faster detection of anomalies than static approaches that rely on fixed intervals. Statistical features allow the adaptive framework to respond more accurately to fluctuations in sensor data, thereby reducing false positives and negatives. Correlation features improve the system's ability to detect coordinated attacks that static systems may fail to recognize.

As shown in Table 7, the adaptive framework demonstrates practical relevance, especially in environments where rapid response and energy efficiency are crucial. The direct comparison shows that the adaptive framework outperforms the static approach in each metric evaluated. Regarding response times, the adaptive framework responds to threats in an average of 250 to 400 milliseconds, depending on the type of attack. In contrast, the static approach requires between 450 and 600 milliseconds, representing a significant difference in environments where response time is critical to mitigate real-time threats. This comparison identifies the limitations of static systems, which struggle to adapt to real-time changes, in contrast to the adaptive framework's ability to reconfigure dynamically in response to emerging threats.

TABLE 7. Comparison between adaptive framework and static approaches.

Security Approach	Response Time (ms)	Adaptability (seconds)	Precision (F1-score)	False Positives (%)	False Negatives (%)
Adaptive Framework	250	3.5	0.92	4	6
Static Approach	450	N/A	0.85	8	12

The adaptive framework demonstrates a substantial advantage in terms of adaptive capability. The system can reconfigure within an average of 3.5 seconds after a threat is detected, ensuring the system continues operating without interruption. Static approaches do not offer adaptive capability, exposing them to recurring or evolving threats requiring a dynamic response. This comparison highlights how static systems are inherently limited by their fixed configurations. This makes them vulnerable to novel or changing attack vectors, unlike the adaptive framework that continuously optimizes its security posture.

On the other hand, threat detection precision is another critical parameter where the adaptive framework stands out. The adaptive framework consistently maintains a precision above 0.9 in most scenarios, while the static approach ranges between 0.78 and 0.85. This improvement in precision is due to the system's ability to adjust its detection algorithms in real time and adapt to threats' characteristics as they evolve, providing a more reliable detection system.

E. QUANTITATIVE RESULTS AND VISUALIZATIONS

The adaptive framework's evaluation is based on several key metrics that allow its performance to be measured compared to traditional or static approaches. These metrics include response time, precision, and false positive and false negative rates. In addition, the evolution of energy consumption and the system's adaptive capacity over time were evaluated. This section presents quantitative results supported by clear visualizations that facilitate their interpretation.

Table 8 shows the exact values of the main evaluation metrics, allowing a direct comparison between the two approaches. The adaptive framework performs in all metrics, with an average response time of 250 ms, compared to 450 ms for the static approach. This represents a significant improvement in terms of operational efficiency, which is crucial for mitigating real-time threats. The precision of the adaptive framework is also superior, reaching 92%, while the static approach remains at 85%. The adaptive approach's false positive and false negative rates are considerably lower, suggesting it can identify and respond to threats more accurately.

TABLE 8. Comparison of metrics between adaptive framework and static approach.

Evaluation Metrics	Adaptive Framework	Static Approach
Response Time (ms)	250	450
Precision (%)	92	85
False Positives (%)	4	8
False Negatives (%)	6	12

Figure 7 clearly and quantitatively represents two critical aspects of the adaptive framework's performance. On the left, the bar chart shows the response times under different types of threats, while on the right, the line chart presents the evolution of the system's energy consumption and adaptive capacity over time.

The bar chart shows how the adaptive framework handles various types of threats. Each bar represents the system's average time to detect and respond to a specific threat. For threats such as network interference, the response time is relatively low (around 250 ms), while for more complex threats such as system overload or false data injection, response times increase, reaching up to 400 ms. This increase in response time reflects the greater complexity of these threats, which require a deeper reconfiguration of the system. Despite this, the adaptive framework can still maintain acceptable response times, which are lower than traditional approaches that are generally much more affected by the complexity of the threats.

Conversely, the line chart details two critical aspects of the system's behavior in real-time: energy consumption and adaptive capacity. As the system responds to threats and adjusts its configuration, a progressive increase in power consumption is observed, rising from 100 mAh at the start of operation to approximately 160 mAh at the end of the analyzed period. This trend reflects the increased resource usage required to maintain system security in threat situations. However, this increase in power consumption is in line with the dynamic adaptive capacity of the framework, justifying the additional energy expenditure to ensure service continuity.

Adaptive capacity, also represented in the line graph, shows minor fluctuations compared to power consumption. This suggests that the system can quickly adjust to threats, with adaptation times varying between 2 and 6 seconds. This adaptive capacity reflects how the system reorganizes its internal components to mitigate threats and ensure the security of the IoT environment. Furthermore, it is observed that the system improves its adaptive capacity for recurring or continuous threats over time, confirming the framework's efficiency in autonomously managing threats.

Figure 8 presents a performance comparison diagram that visualizes the main metrics evaluated between the adaptive framework and the static approach. The results offer a direct and quantifiable view of the improvements that the adaptive framework provides compared to traditional methods, specifically regarding response time, precision, false positives, and false negatives.

Response time is a critical metric in IoT systems, particularly in security environments, where a quick response can mean the difference between containing a threat or allowing it to cause a significant impact. According to the results, the adaptive framework reduces response time to an average of 250 ms, compared to 450 ms for the static approach. This represents a 44% improvement, crucial when reaction speed defines the ability to mitigate damage before the threat compromises the system's integrity.

Regarding precision, the adaptive framework also outperforms the static approach, reaching 92% versus 85% for the traditional approach. This seven-percentage point increase reflects the framework's ability to correctly identify threats without generating erroneous alerts. This directly relates to the false positive and false negative rates, critical metrics in evaluating security systems. False positives, representing security events incorrectly identified as threats, are significantly lower in the adaptive approach, at 4%, versus 8% for the static approach. Similarly, false negatives, representing threats the system fails to detect, are lower in the adaptive framework, at 6%, compared to 12% for the static approach.

These differences are particularly relevant when considering IoT environments, where precision and speed in threat detection are essential for the continuity of operations and the protection of resources. The adaptive framework's ability to deliver better precision and lower error rates while reducing response time makes it a significantly more effective solution for security in IoT systems.

V. DISCUSSION

The results obtained in this study align with previous findings in the IoT security literature but provide essential advances by overcoming the limitations identified in traditional approaches. The studies by Oh and Kim [6], and Kumar et al. [42] emphasize integrating adaptability into IoT security systems, a component not fully developed in their proposals. Hazman et al. [44] showed that static approaches suffer under complex threats, especially regarding response time, which aligns with our findings. Nevertheless, the adaptive framework proposed in this study has significantly reduced response times to an average of 250 ms, demonstrating substantial improvements over the 450 ms reported by traditional static approaches. This reduction is especially crucial for the real-time management of critical IoT systems, where every millisecond counts in threat mitigation.

Our work followed a rigorous evaluation of the adaptive framework under different threat conditions, which allowed us to assess response times, precision, and adaptive capacity.

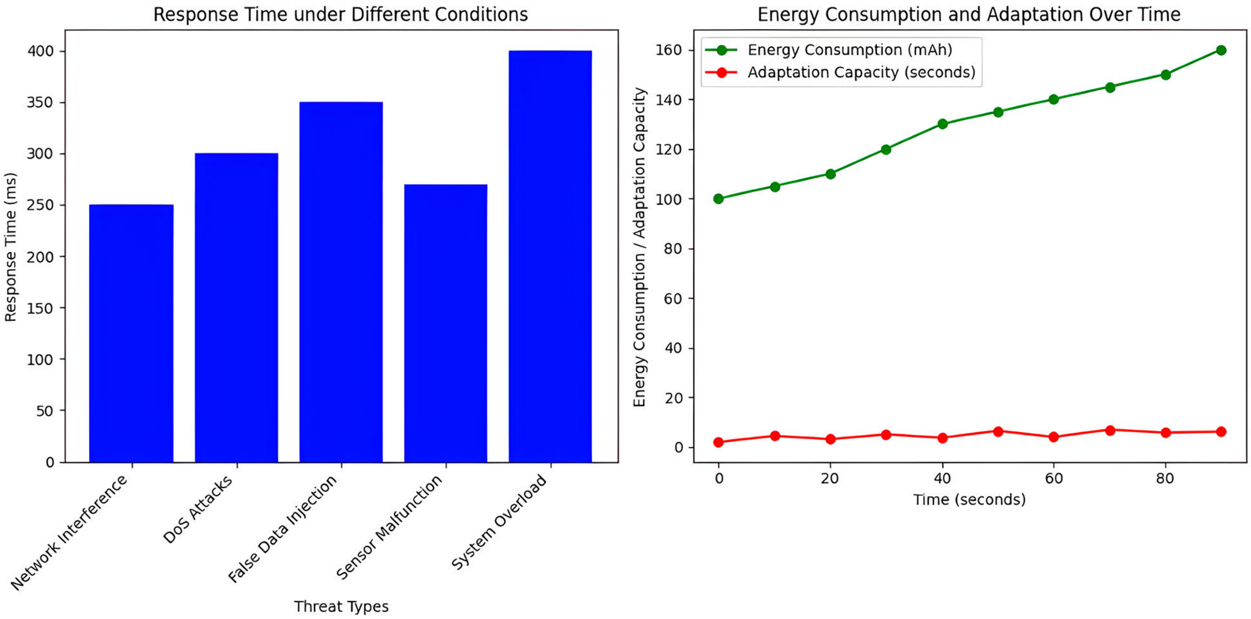

FIGURE 7. Performance comparison in response times, precision, energy consumption and adaptability.

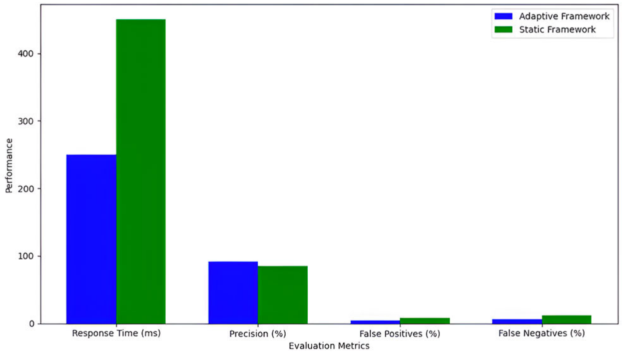

FIGURE 8. Comparative performance diagram of the adaptive framework vs traditional approaches.

Through multi-scale simulations and the analysis of various threats, such as DoS attacks and false data injection, it was possible to observe how the framework dynamically reconfigured itself to mitigate threats while keeping power consumption under control. This methodological approach provides a comprehensive view of system performance, integrating simulations covering individual device levels and the IoT infrastructure. However, deployments in a natural environment may introduce additional complexities, such as heterogeneous network types and varied device capabilities, which could affect the framework's adaptability and resource efficiency.

The results show that the adaptive framework not only improves in terms of speed and precision but also minimizes false positives and false negatives. Compared to traditional approaches, which achieve false positive and negative rates of 8% and 12%, our framework reduces these values to 4% and 6%, as evidenced in Table 8. This is crucial to avoid unnecessary alerts affecting overall system performance, especially in distributed environments where alert overload can lead to system saturation or neglect of actual incidents.

One of the most innovative aspects of this work is the framework's ability to balance energy efficiency and adaptive security. Often, systems that prioritize threat detection precision require increased use of energy resources, resulting in a significant increase in energy consumption. However, the rise in energy consumption of the adaptive framework is gradual and controlled, reaching a maximum of 160 mAh after facing multiple threats without compromising system security. This positions our approach as a viable solution for IoT environments where energy is a limited resource, offering a substantial improvement compared to previous studies, such as that of Kumar et al. [42], which sacrificed precision to achieve energy efficiency.

Despite these promising results, it is essential to consider some limitations inherent to the study. First, the framework was evaluated in a controlled environment, which might not capture all the complexities of a natural IoT environment, such as multiple types of networks, fluctuations in connection quality, or electromagnetic interference not modeled in the simulations. While the results obtained in the laboratory show a clear improvement compared to traditional approaches, the applicability of the framework in natural industrial environments or large-scale IoT networks might require additional adjustments. For example, the observed reconfiguration time may vary if the framework faces a more complex network infrastructure or IoT devices with limited processing capabilities.

Furthermore, assumptions made in the simulation model could have affected the results in specific scenarios. The system was assumed to have constant availability of computational resources to execute security adaptations, which might not be the case in more energy-constrained or resource-limited IoT systems. If the framework were deployed on low-power IoT nodes, the adaptive capacity might be reduced, or additional optimizations might be required to accommodate these limitations. These factors could impact the framework's applicability to IoT environments with more severe energy constraints or complex network configurations.

The interpretation of the results should also consider that the framework was designed for predefined threat scenarios. Although the system showed a remarkable ability to reconfigure itself in response to known threats such as DoS attacks or false data injection, it must investigate how it would behave against emerging threats or new variations of cyberattacks. The adaptive capacity might need adjustments based on the emergence of new attack vectors not contemplated in the current simulations.

The practical relevance of the proposed framework can be illustrated through a scenario in an innovative city environment, where thousands of interconnected IoT devices manage critical infrastructure such as traffic control, energy distribution, and public safety systems. In such a scenario, the dynamic reconfiguration capability of the adaptive framework would be vital in mitigating large-scale attacks, such as DDoS attacks, which could cripple traffic systems or power grids. Unlike static systems requiring manual intervention or taking longer to respond, the adaptive framework could autonomously detect the threat, reconfigure communication paths, and allocate resources efficiently to maintain service continuity. This scenario demonstrates how the framework's ability to respond to threats and optimize resource usage quickly could prevent catastrophic failures in real-world, high-stakes IoT environments.

Although this study demonstrates the efficiency of the adaptive security framework, several open research questions (ORQ) remain that could guide future research and development. Future work should explore how the framework scales in extensive IoT networks, such as smart cities or industrial IoT environments. The scalability of adaptive reconfiguration mechanisms, including the system's ability to handle thousands of devices, must be evaluated to ensure that the framework remains effective and efficient in larger deployments. Additionally, while the framework was effective against predefined threat scenarios, future research should address its adaptability to novel and unpredictable attack vectors. Techniques such as machine learning for zero-day threat detection or anomaly detection in real-time would enhance the framework's flexibility and improve its readiness for evolving cyberattacks.

Further investigations are required to optimize the framework for more energy-constrained environments. Exploring advanced energy-saving algorithms and more efficient hardware-software co-design strategies will be necessary to maintain security while further reducing power consumption in critical IoT devices. To validate the framework's practical utility, future studies must test it in real-world IoT environments where conditions such as network fluctuations, environmental interference, and heterogeneous device capabilities could impact its performance. These tests will help fine-tune the framework to handle the complexities of actual deployments.

Future iterations of this framework could benefit from integrating AI-driven predictive mechanisms to anticipate potential threats before they occur. Exploring reinforcement learning or other advanced AI techniques would allow the system to evolve, improving detection rates and response times while controlling energy consumption.

VI. CONCLUSION

The proposed adaptive security framework was thoroughly evaluated against various threat scenarios, including DoS attacks, false data injection, sensor malfunctions, and system overloads. The results underscore the framework's significant contributions in addressing the evolving challenges in IoT security, where traditional static methods fall short.

The theoretical and practical implications of this research are considerable. The adaptive framework advances the theoretical understanding of dynamic security systems. It proves its practical utility in real-time applications, such as healthcare, industrial monitoring, and smart cities, where high precision and rapid response are essential. One of the primary achievements of this framework is its remarkable reduction in response time—averaging 250 ms, a 44% improvement over static systems, which typically require over 450 ms. This enhanced responsiveness is crucial in IoT environments where even minor delays can lead to significant security breaches or operational failures, especially in time-sensitive applications such as healthcare or industrial monitoring.

Regarding threat detection precision, the adaptive framework exhibited a 92% precision rate, far exceeding the 85% typically achieved by static systems. This performance, accompanied by reduced false positives (4%) and false negatives (6%), highlights the framework's delivery of more accurate threat detection without overwhelming the system with unnecessary alerts. This is a critical feature in large-scale IoT deployments, where alert fatigue can hinder effective threat response.

Practical advantages of this research include its ability to enhance system security while significantly optimizing energy usage. The framework demonstrated that high levels of protection could be maintained with minimal energy consumption, making it highly suitable for deployment in resource-constrained environments, such as remote sensor networks or battery-powered IoT devices. This balance between security and efficiency ensures the framework can be practically implemented in real-world scenarios where both factors are critical. One of the framework's most important aspects is its capacity to optimize energy consumption without compromising security. The framework demonstrated controlled energy use, reaching a maximum of 160 mAh even under multiple threats. This result highlights its potential for deployment in energy-constrained environments, such as sensor networks in remote locations or wearable IoT devices, where power efficiency is just as vital as security.

Beyond these technical advances, the proposed flex framework's adaptability positions it as a robust solution for modern IoT environments. It dynamically reconfigures its security measures based on real-time threat analysis, ensuring continuous protection despite evolving cyberattacks. This adaptability marks a clear departure from the rigid, predefined configurations of static systems, making the proposed solution more suited to the unpredictable nature of today's threats.

However, like any study, this research has limitations. One of the primary limitations is that the framework was tested in a controlled environment, which may not capture all the complexities of real-world IoT ecosystems. Additionally, while the framework was effective against known threats, its ability to detect and adapt to entirely novel or emerging attack vectors remains to be fully validated. These limitations highlight areas for future research and development.

Future research could explore the following directions: First, the framework should be tested and scaled in more extensive and heterogeneous IoT networks, such as smart cities or critical infrastructure environments, to evaluate its performance under more complex conditions. Second, integrating AI-driven predictive mechanisms could enhance the system's capacity to anticipate threats before they fully materialize, adding another layer of security. Finally, further work is needed to refine the balance between security and energy consumption, particularly for long-term deployments in resource-constrained IoT devices.

REFERENCES

[1] I. Kotenko and D. Levshun, "Anomaly detection in IoT networks based on intelligent security event correlation," in *Proc. 16th Int. Conf. Commun. Syst. Netw. (COMSNETS)*, Jan. 2024, pp. 816–824, doi: 10.1109/COMSNETS59351.2024.10426939.

[2] M. Raza Naqvi, M. Waseem Iqbal, M. Usman Ashraf, S. Ahmad, A. T. Soliman, S. Khurram, M. Shafiq, and J.-G. Choi, "Ontology driven testing strategies for IoT applications," *Comput., Mater. Continua*, vol. 70, no. 3, pp. 5855–5869, 2022, doi: 10.32604/cmc.2022.019188.

[3] W. Villegas-Ch, J. García-Ortiz, and S. Sánchez-Viteri, "Toward intelligent monitoring in IoT: AI applications for real-time analysis and prediction," *IEEE Access*, vol. 12, pp. 40368–40386, 2024.

[4] R. Elsayed, R. Hamada, M. Hammoudeh, M. Abdalla, and S. A. Elsaid, "A hierarchical deep learning-based intrusion detection architecture for clustered Internet of Things," *J. Sensor Actuator Netw.*, vol. 12, no. 1, p. 3, 2023, doi: 10.3390/jsan12010003.

[5] M. A. Lawal, R. A. Shaikh, and S. R. Hassan, "An anomaly mitigation framework for IoT using fog computing," *Electronics*, vol. 9, no. 10, p. 1565, Sep. 2020, doi: 10.3390/electronics9101565.

[6] S.-R. Oh and Y.-G. Kim, "Security requirements analysis for the IoT," in *Proc. Int. Conf. Platform Technol. Service (PlatCon)*, Feb. 2017, pp. 1–6, doi: 10.1109/PlatCon.2017.7883727.

[7] B. Velichkovska, A. Cholakoska, and V. Atanasovski, "Machine learning based classification of IoT traffic," *Radioengineering*, vol. 32, no. 2, pp. 256–263, Jun. 2023, doi: 10.13164/re.2023.0256.

[8] H. A. Pham, T. Soriano, V. H. Ngo, and V. Gies, "Distributed adaptive neural network control applied to a formation tracking of a group of low-cost underwater drones in hazardous environments," *Appl. Sci.*, vol. 10, no. 5, p. 1732, Mar. 2020, doi: 10.3390/app10051732.

[9] Y. Y. Ghadi, T. Mazhar, T. A. Shloul, T. Shahzad, U. A. Salaria, A. Ahmed, and H. Hamam, "Machine learning solutions for the security of wireless sensor networks: A review," *IEEE Access*, vol. 12, pp. 12699–12719, 2024, doi: 10.1109/ACCESS.2024.3355312.

[10] G. Z. Ziyatbekova, S. U. Aralbayev, and P. P. Kisala, "Security issues of containerization of microservices," *KazUTB*, vol. 4, no. 21, pp. 1–7, 2023, doi: 10.58805/kazutb.v.4.21-198.

[11] N. A. A. Hussain, S. S. A. Ali, P. Ridao, P. Cieslak, and U. M. Al-Saggaf, "Implementation of nonlinear adaptive U-model control synthesis using a robot operating system for an unmanned underwater vehicle," *IEEE Access*, vol. 8, pp. 205685–205695, 2020, doi: 10.1109/ACCESS.2020.3037122.

[12] M. Thalor and Y. Gharat, "A proposed healthcare architecture using cloud computing in WSN environment with a case study," *Int. J. Integr. Sci. Technol.*, vol. 2, no. 1, pp. 37–44, Jan. 2024, doi: 10.59890/ijist.v2i1.1288.

[13] T. Polonelli, D. Brunelli, A. Marzocchi, and L. Benini, "Slotted Aloha on LoRaWAN-design, analysis, and deployment," *Sensors*, vol. 19, no. 4, p. 838, Feb. 2019, doi: 10.3390/s19040838.

[14] W. A. Jabbar, T. Subramaniam, A. E. Ong, M. I. Shu'Ib, W. Wu, and M. A. de Oliveira, "LoRaWAN-based IoT system implementation for long-range outdoor air quality monitoring," *Internet Things*, vol. 19, Aug. 2022, Art. no. 100540, doi: 10.1016/j.iot.2022.100540.

[15] P. Spadaccino, F. G. Crinó, and F. Cuomo, "LoRaWAN behaviour analysis through dataset traffic investigation," *Sensors*, vol. 22, no. 7, p. 2470, Mar. 2022, doi: 10.3390/s22072470.

[16] A. Adel, "Utilizing technologies of fog computing in educational IoT systems: Privacy, security, and agility perspective," *J. Big Data*, vol. 7, no. 1, p. 99, 2020, doi: 10.1186/s40537-020-00372-z.

[17] D. Palmer, S. Fazzari, and S. Wartenberg, "Defense systems and IoT: Security issues in an era of distributed command and control," in *Proc. Int. Great Lakes Symp. VLSI (GLSVLSI)*, May 2016, pp. 175–179, doi: 10.1145/2902961.2903038.

[18] M. Jayabalan, "Towards an approach of risk analysis in access control," in *Proc. 13th Int. Conf. Develop. eSystems Eng. (DeSE)*, Dec. 2020, pp. 287–292, doi: 10.1109/DeSE51703.2020.9450772.

[19] M. Zeeshan, Q. Riaz, M. A. Bilal, M. K. Shahzad, H. Jabeen, S. A. Haider, and A. Rahim, "Protocol-based deep intrusion detection for DoS and DDoS attacks using UNSW-NB15 and bot-IoT data-sets," *IEEE Access*, vol. 10, pp. 2269–2283, 2022, doi: 10.1109/ACCESS.2021.3137201.

[20] M. L. Hernandez-Jaimes, A. Martinez-Cruz, and K. A. Ramírez-Gutiérrez, "A machine learning approach for anomaly detection on the Internet of Things based on locality-sensitive hashing," *Integration*, vol. 96, May 2024, Art. no. 102159, doi: 10.1016/j.vlsi.2024.102159.

[21] A. Algarni and V. Thayananthan, "Autonomous vehicles: The cybersecurity vulnerabilities and countermeasures for big data communication," *Symmetry*, vol. 14, no. 12, p. 2494, Nov. 2022, doi: 10.3390/sym14122494.

[22] M. A. Khan, R. N. B. Rais, O. Khalid, and S. Ahmad, "Trust-based optimized reporting for detection and prevention of black hole attacks in low-power and lossy green IoT networks," *Sensors*, vol. 24, no. 6, p. 1775, Mar. 2024, doi: 10.3390/s24061775.

[23] M. R. Ghaderi and N. Amiri, "LoRaWAN sensor: Energy analysis and modeling," *Wireless Netw.*, vol. 30, no. 2, pp. 1013–1036, Feb. 2024, doi: 10.1007/s11276-023-03542-y.

[24] R. Marini, K. Mikhaylov, G. Pasolini, and C. Buratti, "LoRaWANSim: A flexible simulator for LoRaWAN networks," *Sensors*, vol. 21, no. 3, p. 695, Jan. 2021, doi: 10.3390/s21030695.

[25] S. S. Narasimha, D. M. Anna, M. N. Vijayalakshmi, and K. S. Raju, "Enabling lightweight device authentication in message queuing telemetry transport protocol," *IEEE Internet Things J.*, vol. 11, no. 9, pp. 15792–15807, May 2024, doi: 10.1109/JIOT.2024.3349394.

[26] K. T. M. Tran, A. X. Pham, N. P. Nguyen, and P. T. Dang, "Analysis and performance comparison of IoT message transfer protocols applying in real photovoltaic system," *Int. J. Netw. Distrib. Comput.*, vol. 12, no. 1, pp. 131–143, Jun. 2024, doi: 10.1007/s44227-024-00021-4.

[27] M. Mishra and S. R. N. Reddy, "Performance assessment and comparison of lightweight D2D-IoT communication protocols over resource constraint environment," *Multimedia Tools Appl.*, vol. 83, no. 26, pp. 67569–67598, Jan. 2024, doi: 10.1007/s11042-024-18132-z.

[28] Á. Michelena, J. Aveleira-Mata, E. Jove, M. Bayón-Gutiérrez, P. Novais, O. F. Romero, J. L. Calvo-Rolle, and H. Aláiz-Moretón, "A novel intelligent approach for man-in-the-middle attacks detection over Internet of Things environments based on message queuing telemetry transport," *Expert Syst.*, vol. 41, no. 2, 2024, Art. no. e13263, doi: 10.1111/exsy.13263.

[29] J. Song, S. Lee, D. Karagiannis, and M. Lee, "Process algebraic approach for probabilistic verification of safety and security requirements of smart IoT (Internet of Things) systems in digital twin," *Sensors*, vol. 24, no. 3, p. 767, Jan. 2024, doi: 10.3390/s24030767.

[30] G. Mao, Y. Liu, W. Dai, G. Li, Z. Zhang, A. H. F. Lam, and R. C. C. Cheung, "REALISE-IoT: RISC-V-based efficient and lightweight public-key system for IoT applications," *IEEE Internet Things J.*, vol. 11, no. 2, pp. 3044–3055, Jan. 2024, doi: 10.1109/JIOT.2023.3296135.

[31] G. Darwesh, J. Hammoud, and A. A. Vorobeva, "A novel approach to feature collection for anomaly detection in kubernetes environment and agent for metrics collection from kubernetes nodes," *Sci. Tech. J. Inf. Technol., Mech. Opt.*, vol. 23, no. 3, pp. 538–546, Jun. 2023, doi: 10.17586/2226-1494-2023-23-3-538-546.

[32] L. Maier, D. Jansen, F. Wüllhorst, M. Kremer, A. Kümpel, T. Blacha, and D. Müller, "AixLib: An open-source modelica library for compound building energy systems from component to district level with automated quality management," *J. Building Perform. Simul.*, vol. 17, no. 2, pp. 196–219, Mar. 2024, doi: 10.1080/19401493.2023.2250521.

[33] C. Kotronis, M. Nikolaidou, A. Tsadimas, C. Michalakelis, and D. Anagnostopoulos, "Extending SysML to integrate cost analysis into model-based systems engineering," *IEEE Trans. Eng. Manag.*, vol. 71, pp. 2865–2880, 2022, doi: 10.1109/TEM.2022.3200148.

[34] E. B. Sanjuan, I. A. Cardiel, J. A. Cerrada, and C. Cerrada, "Message queuing telemetry transport (MQTT) security: A cryptographic smart card approach," *IEEE Access*, vol. 8, pp. 115051–115062, 2020, doi: 10.1109/ACCESS.2020.3003998.

[35] P. Zhang, "IEEE draft standard for spectrum characterization and occupancy sensing," *IEEE P802.22/D5*, pp. 1–89, Jul. 2019.

[36] H.-B. Tran and V. T.-A. Phan, "Potential usage of fly ash and nano silica in high-strength concrete: Laboratory experiment and application in rigid pavement," *Case Stud. Construct. Mater.*, vol. 20, Jul. 2024, Art. no. e02856, doi: 10.1016/j.cscm.2024.e02856.

[37] D. Soler, I. Cillero, C. Dafonte, M. Fernández-Veiga, A. F. Vilas, and F. J. Nóvoa, "QKDNetSim+: Improvement of the quantum network simulator for NS-3," *SoftwareX*, vol. 26, May 2024, Art. no. 101685, doi: 10.1016/j.softx.2024.101685.

[38] F. Ershadi, M. Nazari, and M. S. Chegenie, "Native speakerism as a source of agency-related critical incidents: Implications for non-native English teachers' professional identity construction," *System*, vol. 120, Feb. 2024, Art. no. 103182, doi: 10.1016/j.system.2023.103182.

[39] V. Mittal and S. Gillespie, "Using model-based systems engineering to avoid unnecessary technology resulting from dynamic requirements," *IEEE Trans. Eng. Manag.*, vol. 71, pp. 2660–2671, 2022, doi: 10.1109/TEM.2022.3181268.

[40] A. B. Kathole, K. N. Vhatkar, and S. D. Patil, "IoT-enabled pest identification and classification with new meta-heuristic-based deep learning framework," *Cybern. Syst.*, vol. 55, no. 2, pp. 380–408, Feb. 2024, doi: 10.1080/01969722.2022.2122001.

[41] N. Y. Fares, D. Nedeljkovic, and M. Jammal, "AI-enabled IoT applications: Towards a transparent governance framework," in *Proc. IEEE Global Conf. Artif. Intell. Internet Things (GCAIoT)*, Dec. 2023, pp. 109–114, doi: 10.1109/GCAIoT61060.2023.10385106.

[42] P. Kumar, H. Bagga, B. S. Netam, and V. Uduthalapally, "SAD-IoT: Security analysis of DDoS attacks in IoT networks," *Wireless Pers. Commun.*, vol. 122, no. 1, pp. 87–108, Jan. 2022, doi: 10.1007/s11277-021-08890-6.

[43] D. Schmidt, C. Tagliaro, K. Borgolte, and M. Lindorfer, "IoTFlow: Inferring IoT device behavior at scale through static mobile companion app analysis," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2023, pp. 681–695, doi: 10.1145/3576915.3623211.

[44] C. Hazman, A. Guezzaz, S. Benkirane, and M. Azrour, "LIDS-SIoEL: Intrusion detection framework for IoT-based smart environments security using ensemble learning," *Cluster Comput.*, vol. 26, no. 6, pp. 4069–4083, Dec. 2023, doi: 10.1007/s10586-022-03810-0.

WILLIAM VILLEGAS-CH (Member, IEEE) received the master's degree in communications networks and the Ph.D. degree in computer science from the University of Alicante. He is currently a Professor of information technology with Universidad de Las Américas, Quito, Ecuador. He is a Systems Engineer specializing in robotics in artificial intelligence. He has participated in various conferences as a speaker on topics, such as ICT in education and how they improve educational quality and student learning. His main articles focus on the design of ICT systems and models and prototypes applied to different academic environments, especially with the use of big data and artificial intelligence as a basis for creating intelligent educational environments. His main research interests include web applications, data mining, and e-learning.

ROMMEL GUTIERREZ received the master's degree in cybersecurity. He is currently a Research Technician with UDLA, Quito, Ecuador, where he applies his knowledge in software development, data science, and cybersecurity. As an IT Engineer, his focus on AI, data science, cybersecurity, and software development is particularly geared toward education and research. He's passionate about utilizing these technological tools to fortify digital systems and create innovative solutions, with a special emphasis on their applicability in educational settings and research environments.

IVÁN SÁNCHEZ-SALAZAR (Member, IEEE) received the Engineering degree from the National Polytechnic School, Ecuador, in 2006, the M.Eng. degree from the Central University of Ecuador, in 2015, and the Ph.D. degree in telecommunications engineering from the University of Málaga, Spain, in 2024. He is currently a Professor with Universidad de Las Américas, Ecuador. His research interests include digital communications, electronic circuits, electromagnetic theory, and data acquisition with the IoT devices.

ARACELY MERA-NAVARRETE received the master's degree in business administration from UIDE. She is currently a Computer Engineer in Quito, Ecuador. She is an Expert in E-learning platforms FATLA.ORG, her skills and abilities are in computer science and its associated technologies, such as hardware, software, communications, e-learning platforms, construction of computer systems, and management in LMS applications (Moodle-CANVAS).