Evaluating AES Payload Encryption for Securing MQTT-based Smart Home Networks with Machine Learning-based Intrusion Detection

Mariusz Gajewski[1] and Wojciech Sałabun[1,2]

[1]*National Institute of Telecommunications, Warsaw, Poland,*
[2]*West Pomeranian University of Technology, Szczecin, Poland*

https://doi.org/10.26636/jtit.2026.2.2545

Abstract — The message queuing telemetry transport (MQTT) protocol is widely adopted in smart home IoT ecosystems despite its default configuration failing to offer adequate protection against eavesdropping or payload manipulation. This study addresses an important research gap and attempts to determine whether AES-128 payload encryption is capable of securing MQTT transmissions without degrading the effectiveness of machine learning-based intrusion detection systems (IDS). Three security configurations, namely TLS, payload encryption, and token-based authentication, deployed on the ESP32 microcontroller family, are compared and their impact on message latency is measured. Experimental results show that the AES-128 encryption overhead remains at below 25% of the message publication time on ESP32-S3. To evaluate the robustness of IDS under encryption, we apply a reproducible modification to the MQTTset benchmark dataset that replaces variable-length plaintext payloads with fixed-length ciphertext representations while simultaneously preserving feature semantics and labeling consistency. 5 out of 6 evaluated classifiers maintained their accuracy level at above 99% on the modified dataset, with tree-based and neural models showing no meaningful degradation. Only Naive Bayes proved unsuitable, with its accuracy dropping from 98.79% to 62.15% due to its independence assumptions being violated by cryptographic uniformity. These results confirm that AES-based MQTT payload encryption is a practical and efficient security measure for resource-constrained IoT environments, provided that appropriate classifiers are employed.

Keywords — AES, encryption, feature selection, machine learning, MQTT

1. Introduction

The proliferation of interconnected devices in home networks relies increasingly on standard communication protocols such as MQTT. This enables the integration of diverse devices, simultaneously facilitating efficient data aggregation and distribution. However, such connectivity raises significant security concerns, particularly given the limitations of available bandwidth, computing power, memory usage, and power supply constraints affecting the devices concerned. Although they are typically not targeted as frequently as enterprise infrastructures, home networks remain vulnerable to manipulation consisting, for instance, in impersonating a valid network endpoint, making data encryption essential. While gaining in popularity and being applied on a wide scale, the message queuing telemetry transport (MQTT) protocol does not provide built-in encryption at the application layer [1], [2].

Despite extensive research on securing MQTT communication and on machine learning-based intrusion detection systems (IDSs), interaction between MQTT payload encryption and protocol-level attack detectability remains insufficiently explored. In particular, it remains unclear whether encrypting application-layer data alters the statistical properties of MQTT traffic in a way that degrades the effectiveness of IDSs trained on protocol features. This uncertainty is especially critical for smart home environments, where resource-constrained devices must balance security, performance, and detection capability.

Most recent studies either assume plaintext payloads for IDS design or focus exclusively on cryptographic protection without validating its impact on anomaly detection. In contrast, this work explicitly investigates whether AES-based MQTT payload encryption may be deployed without sacrificing IDS performance, combining embedded-system measurements with a modified benchmark dataset to ensure fair and reproducible evaluation.

1.1. Research Problem and Objectives

Securing MQTT communication in smart home environments poses a fundamental design challenge. On the one hand, resource-constrained devices such as ESP32-based sensors have limited computational resources, making complex cryptographic mechanisms impractical. On the other hand, plaintext MQTT payloads are trivially susceptible to eavesdropping and manipulation by any party with access to the local network segment.

Transport layer security (TLS) addresses confidentiality at channel level but does not protect payload data end-to-end, e.g. when messages are relayed through an intermediary broker or stored for later processing. Lightweight payload-level

encryption, such as AES-128, may complement TLS by providing end-to-end content protection at a potentially lower computational cost. However, whether this added encryption layer is practically feasible on resource constrained microcontrollers, and at what latency cost, remains an open empirical question.

An equally critical concern arises at the intersection of encryption and intrusion detection. Machine learning-based IDSs trained on MQTT traffic typically rely on features derived from protocol-level fields, including packet lengths, timing patterns, and message types. When payload encryption transforms variable-length plaintext into fixed-length ciphertext, it alters the statistical distribution of these features. Therefore, it is unclear whether classifiers trained on plaintext traffic will retain their detection accuracy when applied to encrypted traffic. Existing benchmark datasets, such as MQTTset [3], contain plaintext payloads only and thus cannot be used directly to evaluate IDS robustness under encryption. A systematic methodology for adapting such datasets to reflect encrypted payloads, while preserving feature semantics and label consistency, is currently unavailable.

1.2. Contribution of This Work

This work addresses the problem of securing MQTT-based smart home communication while preserving the effectiveness of machine learning-based intrusion detection systems operating on protocol-level features. Unlike many existing studies that focus either on cryptographic protection or on intrusion detection in isolation, this paper considers their interaction under realistic resource constraints. As a result, the contributions of this study span both scientific insights into IDS behavior under encrypted traffic and engineering validation of lightweight security mechanisms on embedded platforms.

From a methodological and analytical perspective, this paper provides:

- a systematic investigation of how AES-based MQTT payload encryption influences the statistical characteristics of protocol-level features used by machine learning–based intrusion detection systems,
- a reproducible methodology for adapting an established benchmark dataset to reflect encrypted, fixed-length MQTT payloads while preserving feature semantics and labeling consistency,
- a comparative evaluation of classical and neural machine learning classifiers, revealing which algorithmic families remain robust under payload encryption and which are adversely affected by cryptographic transformations.

From an implementation and validation perspective, the paper delivers:

- a practical implementation and comparison of multiple MQTT security configurations, including TLS, authentication mechanisms, and AES-128 payload encryption, on ESP32 microcontrollers representing different processor architectures,
- detailed experimental measurements of connection setup time, message publication latency and encryption overhead under constrained computational and energy conditions,
- an end-to-end validation demonstrating that payload-level encryption can be integrated into MQTT-based smart home systems without compromising intrusion detection capabilities, provided that appropriate classifiers are employed.

The remainder of this paper is structured as follows. Section 2 reviews related work, Section 3 describes the approach developed by the authors and the experimental setup. Section 4 evaluates performance of the encryption method used. Section 5 presents and discusses performance of the proposed ML algorithm-based IDSs, while Section 6 concludes the study.

2. Related Works

As the number of networked devices continues to grow, home network security becomes an increasingly complex issue. This makes home network infrastructure an attractive target for attacks and a platform for launching attacks on other environments (e.g. botnets). A review of the literature describing attacks on home networks reveals a considerable number of scientific works focusing on the topic in question. Integration of existing solutions with monitoring and control systems presents additional challenges. MQTT is a popular solution used in this specific area. MQTT defines how IoT devices can publish and subscribe to data over the Internet.

The sender (publisher) and the receiver (subscriber) communicate via topics and are decoupled from each other. The MQTT broker filters incoming messages and distributes them correctly to the subscribers.

2.1. Home Network Threats Described in the Literature

The survey presented in [4] provides a systematic literature review of cybersecurity in Smart Home Internet of Things (SHIoT) environments. The authors catalog common attacks against SHIoT, including: (i) brute-force attacks, (ii) data breaches and monitoring, (iii) denial-of-service (DoS) attacks, (iv) data forgery, and (v) spam or malware. Paper [5] maps this classification to attacks targeting the MQTT protocol.

When analyzing vulnerabilities and potential attacks on the MQTT protocol, the authors also considered changes introduced in its subsequent versions, particularly the latest 5.0 iteration[1]. They also provide defense strategies and best practices for hardening MQTT-aware nodes. In this context, several approaches are selected:

- proper authentication and robust access control lists (ACLs) to restrict publish/subscribe capabilities per client and topic,
- end-to-end security considerations, including payload-level encryption or application-layer cryptographic protections to complement TLS and achieve true end-to-end confidentiality,

[1] https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

- secure broker and client deployment practices,
- monitoring, anomaly detection, and testing approaches to identify misconfigurations and vulnerable implementations before exploitation.

The authors also emphasize the role of modern techniques such as ML-based anomaly detection and cryptographic mechanisms in mitigating MQTT-specific threats.

Unfortunately, while the literature provides a comprehensive taxonomy of attacks targeting smart home and MQTT-based environments, it does not address how such threats can be effectively detected when application-layer data are protected using payload-level encryption.

2.2. Securing MQTT-based Networks

Effectively securing MQTT-based networks is essential, because MQTT is often used in IoT systems where devices are resource-constrained and widely distributed. Common approaches and best practices for securing MQTT deployments include the following:

- transport-layer encryption (TLS/mTLS),
- authentication and authorization of clients (publishers and subscribers),
- securing the topic structure,
- protecting the payload through encryption and/or digital signatures,
- hardening the software layer of MQTT-based systems and brokers (operating systems, hypervisors, etc.),
- monitoring system behavior and enforcing the principle of least privilege.

There is a substantial body of research addressing these aspects and proposing various security enhancements.

The proposed countermeasures include secure configuration practices for nodes and brokers, intrusion detection systems, and cryptographic mechanisms for protecting both MQTT sessions and data payload. Specifically, article [6] proposes a lightweight symmetric encryption algorithm optimized for MQTT constraints to enhance confidentiality and integrity of device-broker communication. In [7], an in-depth analysis of MQTT protocol security is presented, including experiments with different cryptographic integrations under IoT-specific constraints. In the study, AES-CBC, RSA, and ECC-AES are applied to encrypt the message payload, and the system is tested against attacks such as black-box penetration, man-in-the-middle (MiTM), identity spoofing, and denial-of-service (DoS).

The authors of [8] analyze the overhead associated with double encryption in end-to-end security models and propose improvements for streamlined secure MQTT communication with robust key management and authentication mechanisms. Similarly, a novel approach based on AugPAKE authentication and PRESENT encryption is proposed in [9] to achieve mutual authentication, confidentiality, and non-repudiation in MQTT-based applications.

Another approach is presented in [10], where the MQTT-TLS profile for authentication and authorization for the constrained environments (ACE) framework specified by the IETF [11] is implemented and evaluated. This work focuses on implementing the functionality of the authorization server for client registration, authorization policies and access tokens, as well as broker-side mechanisms enforcing authentication across different MQTT versions.

In contrast to the above approaches which primarily focus on strengthening confidentiality and authentication, the impact of MQTT payload encryption on protocol-level traffic characteristics and subsequent intrusion detection performance has received limited attention.

2.3. Anomaly/attack Detection of MQTT-based Networks

The field of anomaly and attack detection in MQTT-based IoT networks has experienced significant growth, driven by the proliferation of IoT devices and the security vulnerabilities inherent in the lightweight MQTT protocol. Despite its widespread adoption in industrial IoT, smart home, and medical IoT applications, MQTT faces critical security challenges. The protocol lacks encryption and authentication by default, making it vulnerable to various attack.

Previous research has extensively evaluated classical machine learning (ML) algorithms for MQTT attack detection. Studies comparing Naive Bayes, k-nearest neighbors (k-NN), decision trees, random forest, support vector machines, and logistic regression demonstrate varying degrees of effectiveness across different attack types. An overview of the application of classical ML methods for anomaly detection (one-class) and attack classification (multiclass) is provided, among others in [12]–[14]. These studies address the problem of detecting attacks in IoT networks using different communication protocols, including MQTT.

Moreover, deep learning approaches have shown strong potential for detecting MQTT intrusion. In particular, the authors of [15] proposed a deep neural network (DNN) architecture designed specifically for analyzing MQTT traffic, achieving accuracies of 99.92% for uni-flow, 99.75% for bi-flow, and 94.94% for packet-flow binary classification tasks. Their work compares DNN performance with traditional ML algorithms (Naive Bayes, random forest, k-NN, and decision trees) as well as sequence-based models such as LSTM and GRU, demonstrating the advantages of deep learning for MQTT attack detection.

Feature selection techniques, including chi-square statistics, correlation analysis, and wrapper methods, have also been shown to improve model efficiency while maintaining high detection accuracy.

Most studies rely on benchmark datasets containing MQTT traffic. Such datasets have gained popularity in research on IoT intrusion detection systems, because they provide a shared and reproducible basis for developing and evaluating detection methods. Benchmark datasets typically include both legitimate MQTT traffic and well-characterized attack scenarios (e.g., DoS, brute force, and malformed data), providing ground truth labels for supervised learning. Datasets such as MQTTset [3] and MQTTEEB-D [16] were specifically de-

signed to provide rich labeled MQTT flows for supervised and semi-supervised intrusion detection, including protocol-level features and realistic attack scenarios. More recently, MQTT_UAD was published in [17]. It is a publicly available dataset containing MQTT traffic under denial-of-service, man-in-the-middle, and intrusion attacks, enabling reproducible evaluation of classification models in MQTT-specific environments.

At a broader IoT scale, [18] introduces CICIoT2023 – a large-scale benchmark comprising traffic from 105 real IoT devices communicating via MQTT, CoAP, and RTSP protocols, with over 30 attack types spanning DDoS, reconnaissance, spoofing, and brute-force categories. While these datasets significantly advance the availability of labeled IoT traffic for IDS research, none of them includes encrypted MQTT payloads, leaving the question of IDS robustness under payload encryption unanswered.

3. Approach and Experimental Setup

The proposed approach assumes that the chosen security method represents a trade-off between effectiveness (understood as the level of security offered), operational speed, and cost-effectiveness, where cost-effectiveness is understood as energy consumption and computational resource utilization. In MQTT-based home networks, data-providing devices have limited technical capabilities and often rely on battery power supply. Deployment of complex cryptographic mechanisms to protect data transmitted by simple sensors, such as temperature or motion devices, is therefore disproportionate to the threat model. The primary security emphasis is placed on network access control at the wireless layer, while lightweight payload-level encryption is evaluated as a feature.

The general approach consists of two main phases. The former covers the verification of performance indicators for various security configurations and the selection of the optimal solution. The other addresses the mapping of the selected mechanism to protocol parameters at the TCP and MQTT levels, followed by the adaptation of an established benchmark dataset and the development of corresponding evaluation guidelines.

Data transmission times were measured across four scenarios for client-to-broker publication: plaintext transmission, authentication-only transmission, AES-128 encrypted payload transmission, and TLS-secured transmission. Peak current consumption was recorded during each operation to assess energy impact.

Figure 1 illustrates the complete interaction sequence between a publisher and a broker, including the optional data preparation phase introduced by payload encryption. Attack and anomaly detection performance was subsequently evaluated on the modified dataset, using supervised multiclass classifiers trained on protocol-level features, with their accuracy and F1 scores computed on identical train-test splits to ensure comparability with baseline results.

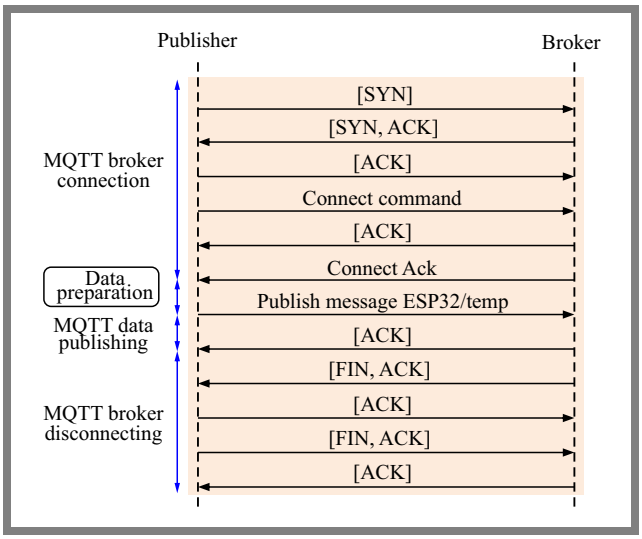

Fig. 1. Connection and MQTT data transmission time spans.

The diagram below outlines the standard process of connection, handshaking, data publishing, and disconnection, highlighting the key message exchanges and acknowledgements involved. The figure also shows an optional data preparation phase, which occurs only when MQTT messages are encrypted. The preparation time was therefore considered only in scenarios involving MQTT payload encryption.

4. Evaluation of Encryption Method Performance

Experimental evaluation was conducted on two microcontrollers from the ESP32 family: ESP32-S3 and ESP32-C3. The ESP32-S3 and ESP32-C3 are based on the Xtensa and RISC-V architectures, respectively. All measurements were conducted with the clock at 80 MHz to ensure comparable power conditions, although the ESP32-S3 supports speeds of up to 160 MHz. Identical library releases were used to implement WLAN, TLS, and MQTT on both devices, and the PSRAM external memory was disabled on the ESP32-S3 to ensure the same testing conditions. Timing measurements were determined for the following publication scenarios:

- plaintext publication without authentication,
- plaintext publication with authentication,
- TLS-secured plaintext publication without authentication,
- TLS-secured plaintext publication with authentication,
- AES-128 encrypted publication without authentication.

All timing measurements were performed directly on the ESP32 microcontrollers. Data were transmitted over Wi-Fi, with the ESP32 connected to a 2.4 GHz access point. The access point and the MQTT broker server were both connected via Ethernet to the same network switch. The ESP32 was located approximately 4 m from the access point, with no obstacles between the devices. Timing was measured using the `micros()` function, providing 1 µs resolution, which is adequate for operations on the millisecond scale. For each scenario, at least 500 measurement repetitions were executed,

Tab. 1. MQTT broker connection time.

Method	ESP32-S3		ESP32-C3	
	Mean [µs]	Std_dev [µs]	Mean [µs]	Std_dev [µs]
Plain	16 082.18	1 512.41	7 722.20	1 123.36
Auth	20 145.21	5 467.32	11 855.32	3 652.25
TLS	2 178 982.40	5 631.76	974 977.46	4 622.74
TLS_auth	2 188 194.09	10 181.12	1 623 887.00	16 832.54
Encrypted	19 160.12	2 737.36	13 171.94	1 729.27

Tab. 2. MQTT data publication time.

Method	ESP32-S3		ESP32-C3	
	Mean [µs]	Std_dev [µs]	Mean [µs]	Std_dev [µs]
Plain	2052.82	9.95	1105.60	4.45
Auth	2053.88	6.35	1105.95	6.41
TLS	1123.40	8.23	1144.50	6.23
TLS_auth	1103.95	7.06	2334.40	131.69
Encrypted	2100.60	8.57	2252.17	169.17

preceded by a series of 50 preliminary runs to minimize cold-start effects and stabilize cache memory contents.

The measurement campaign and MQTT result reporting were implemented using the base Arduino-esp32 library, which internally relies on ESP-IDF framework components for Wi-Fi, TCP/IP, and MQTT operations. AES-128 payload encryption was implemented using the AESLib library [19], a portable software-based AES implementation that operates across multiple hardware platforms and programming languages without relying on platform-specific hardware acceleration.

This choice favors implementation portability and reproducibility over platform-specific optimization. While absolute timing values are platform and implementation-specific, the relative overhead of AES-128 encryption compared with the connection setup and message publication is expected to remain consistent across implementations that use the same standard networking libraries, as the dominant latency contributors (Wi-Fi stack, TLS handshake, TCP operations) are governed by the vendor-provided framework.

For each scenario, mean execution time and standard deviation were recorded. All timing measurements were saved after successful network authentication and broker availability. The results are presented in Tab. 1.

Furthermore, the time required for data publication by the MQTT client running on the ESP32 was determined. The transmitted data consisted of floating-point values with two decimal places. The results are presented in Tab. 2.

Tab. 3. Data preparation (encryption) time.

Method	ESP32-S3		ESP32-C3	
	Mean [µs]	Std_dev [µs]	Mean [µs]	Std_dev [µs]
Plain	0.00	0.00	0.00	0.00
Auth	0.00	0.00	0.00	0.00
TLS	0.00	0.00	0.00	0.00
TLS_auth	0.00	0.00	0.00	0.00
Encrypted	397.00	5.26	495.22	6.47

Finally, the AES-128 encryption time was measured. The encryption process was performed after establishing the MQTT connection and before data publication. If encryption was not required for a given publication scenario, the encryption time was considered as zero.

As shown in Tabs. 1 – 3, the AES-128 encryption time is less than 50% of the publication time on the ESP32-C3 and less than 25% of the publication time on the ESP32-S3. It is also noteworthy that the time required to establish MQTT sessions is significantly longer than the time required for message publication or payload encryption.

Longer connection and publication times were observed for the ESP32-S3 compared to the ESP32-C3. This is likely caused by architectural differences between the two platforms. The ESP32-S3 features a dual-core Xtensa LX7 processor, whereas the ESP32-C3 employs a single-core RISC-V architecture. In the dual-core ESP32-S3, Wi-Fi-related tasks are typically pinned to core 0 by default within the Wi-Fi driver framework. In contrast, components such as the lightweight IP (lwIP) TCP/IP stack, operating at priority level 18, remain unpinned and may execute on either core. This configuration requires inter-core synchronization and task handoffs through the RTOS, which may increase latency in TCP transmission operations.

By contrast, the single-core ESP32-C3 executes all processes on a single core, eliminating inter-core synchronization overhead and potentially reducing networking latency at the same clock speed.

Considering the obtained data, we conclude that MQTT payload encryption ensures integrity and confidentiality of transmitted data within a local network. Assuming that the local Wi-Fi network is properly secured and access requires authentication, encrypting MQTT payload data further enhances communication security, although it increases the size of the transmitted data.

In the following sections, we demonstrate that increasing the payload size to 16 bytes (corresponding to the size of an AES-128 ciphertext for a two-digit floating-point value) does not degrade the effectiveness of machine learning algorithms used for anomaly and attack detection. Therefore, MQTT payload encryption appears to be a practical and promising approach for improving security in smart home networks.

5. Performance of the ML Algorithm-based IDS

We evaluated the detectability of anomalies and attacks in MQTT-based systems using a public MQTTset dataset, available at [2]. This dataset was chosen for the following reasons:

- It defines a typical IoT environment of smart home ecosystems, where data are primarily published by sensor devices. Therefore, the traffic patterns of publishers resemble those observed in smart home deployments. Although the dataset

[2] https://www.kaggle.com/datasets/cnrieiit/mqttset/data

is synthetic – as the network traffic was generated based on sensor behavior models – it has been recognized within the academic community as one of the representative datasets for IoT and smart home research. Additionally, the dataset includes packet capture (PCAP) files, enabling a detailed analysis of MQTT traffic across multiple layers of the OSI model.

- It uses plaintext MQTT communication (via the 1883 port with an MQTT broker), which enables a simple comparison of ML algorithm's performance on both the original and modified datasets under identical conditions.

The dataset MQTT set [3], because its vulnerability simulations focus on common and easily identifiable cyberattacks targeting MQTT traffic:

- *Flooding denial-of-service (DoS)* – aimed at saturating the MQTT broker by establishing multiple client connections and maximizing message transmission within each connection.
- *MQTT Publish flood* – in which a malicious IoT device establishes an MQTT connection and launches a DoS attack by periodically transmitting a lot of MQTT Publish messages within a single connection. The objective is to overload server resources such as connection slots, network bandwidth, thereby blocking normal communication.
- *SlowITe* – a DoS attack targeting the MQTT application protocol that requires minimal bandwidth and resources. It operates by initiating a large number of connections to the MQTT broker to occupy all available connection slots.
- *Malformed data* – generates and sends malformed packets to the broker to trigger exceptions in the target service. In the considered scenario, a sequence of malformed Connect or Publish packets is transmitted to the broker.
- *Brute-force authentication* – involves repeated attempts to guess user credentials used for MQTT authentication. Since DoS and DDoS attacks remain among the most common threats, the dataset reflects their characteristics as well as attacks targeting authorization mechanisms, malformed data, and violations of the MQTT protocol.

A limitation of this dataset is that it does not capture specification of wireless transmission, such as connection establishment delays caused by medium access. However, when focusing on anomaly and attack detection methods based on MQTT protocol-level features, this limitation does not restrict the usefulness of the dataset for further analysis.

Encrypting messages published by MQTT clients results in fixed-length payloads. This modification does not significantly affect anomaly and attack detection results when ML methods rely primarily on protocol-level features. To verify this assumption, we modified the MQTTset dataset accordingly.

As mentioned earlier, the modification involved assigning a new MQTT length value to Publish messages. Because the encrypted MQTT payload has a fixed size of 32 bytes, the MQTT message length depends only on the topic length.

Tab. 4. Effectiveness of classification model (16 bytes).

Algorithm	Accuracy	F1 score	Δ Accuracy	Δ F1 score
Neural network	0.9923	0.9908	-0.0010	-0.0025
Random forest	0.9971	0.9969	0.0028	0.0026
Naive Bayes	0.6215	0.7650	–0.3664	–0.2247
Decision tree	0.9971	0.9969	0.0191	0.0119
Gradient boost	0.9949	0.9947	0.0038	0.0030
Multilayer perceptron	0.9939	0.9936	0.0470	0.0299

Therefore, the modified MQTT length value was defined as the sum of the fixed payload size, the fixed MQTT topic length field (2 bytes), and the variable topic length.

Furthermore, each MQTT payload field was filled with a random string representing a 16-byte ciphertext encoded as 32 characters. This resulted in a modified dataset on which ML classifiers were applied to compare results with those obtained from the original dataset from [3]. Both datasets have the same feature schema, labeling strategy, and preprocessing pipeline to ensure comparability.

The modification was applied to the dataset containing MQTT network traffic extracted from PCAP traces. The dataset also includes related feature vectors enabling it to be used in ML classification tasks. Importantly, the modification affected only the subset representing legitimate traffic, while subsets describing anomalous or attack-related traffic remained unchanged.

For validation, supervised ML classifiers were trained separately on the original and modified datasets using identical architectures, hyperparameters, and train-test splits. Model performance was evaluated using accuracy and F1 score. Additionally, feature importance rankings were analyzed to determine whether enforcing fixed-length payloads altered the discriminative characteristics of the data.

Within the detection framework, the same widely applied multiclass classifiers as used in [3] were employed. In particular, the original study focused on decision tree (optimized), random forest, gradient boosting, multilayer perceptron (MLPC), a neural network implemented using Keras, and Gaussian Naive Bayes algorithms. To ensure comparability of results, we used the same classification algorithms and parameter settings.

Tables 4 and 5 present the classification performance for MQTT communication scenarios where publishers transmit encrypted 16- and 32-byte payloads, respectively.

Model performance was evaluated using accuracy and F1 score metrics. Accuracy represents the percentage of correctly classified instances, while the F1 score combines precision and recall. These metrics are based on the standard confusion matrix components: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

Despite the introduction of payload encryption, 5 of 6 algorithms maintained high classification performance with their

Tab. 5. Effectiveness of classification model (32 bytes).

Algorithm	Accuracy	F1 score	Δ Accuracy	Δ F1 score
Neural network	0.9922	0.9905	-0.0011	-0.0027
Random forest	0.9971	0.9969	0.0028	0.0026
Naive Bayes	0.6215	0.7650	–0.3664	–0.2247
Decision tree	0.9971	0.9969	0.0191	0.0119
Gradient boost	0.9949	0.9947	0.0038	0.0030
Multilayer perceptron	0.9955	0.9952	0.0486	0.0215

accuracy remaining within 5 percentage points of the baseline results. Random forest and decision tree classifiers showed slight performance improvements (accuracy Δ: +0.0028 to +0.0191), achieving accuracy of 99.71% on encrypted payloads. Gradient boosting maintained 99.49% accuracy with minimal variation (accuracy Δ: +0.0038). The neural network model showed only a negligible decrease in performance (accuracy Δ: –0.0010 for both payload lengths), while the multilayer perceptron improved to 99.39% and 99.55% for the 16-byte and 32-byte payload scenarios, respectively (accuracy Δ: +0.0470 and +0.0486).

In contrast, the Naive Bayes classifier exhibited a substantial performance degradation after the introduction of encrypted MQTT payloads. Its accuracy dropped from 98.79% to 62.15% in both scenarios.

This behavior can be explained by the conditional independence assumption underlying the Naive Bayes model, which assumes that features are statistically independent given the class label. AES-128 encryption produces a cryptographically uniform output, which violates these probabilistic assumptions. In contrast, tree-based models such as random forest and decision tree rely on recursive partitioning based on information gain or Gini impurity, making them less sensitive to feature distribution assumptions. Applying encryption to the MQTT payload shifts class separability to different feature subspaces rather than eliminating it, allowing tree-based models to adapt effectively.

Similarly, neural network architectures can learn non-linear feature transformations that generalize across both encrypted and unencrypted traffic patterns. The slight performance improvements observed for some classifiers suggest that cryptographic uniformity may reduce overfitting to noise patterns present in the original unencrypted data. Importantly, the negligible performance differences observed for 5 of 6 classifiers are not a limitation but rather a central finding of this study. The fact that accuracy and F1 score remain virtually unchanged after payload encryption confirms that these classifiers rely predominantly on protocol-level features, such as packet timing, message type distributions, and connection patterns, rather than on payload content.

This observation directly validates the premise underlying our dataset modification methodology. If features are properly selected at the protocol level, encrypting the application-layer payload does not degrade attack detectability. The sole exception, Naive Bayes, serves as a useful negative control, demonstrating that classifiers with strong distributional assumptions can indeed be affected by encryption-induced changes in feature statistics.

6. Conclusions

The use of MQTT payload encryption provides effective protection against eavesdropping and data integrity violations in home sensor networks. As demonstrated in this study, the implementation of the AES-128 algorithm on popular ESP32 microcontroller platforms efficiently handles the data encryption process. Moreover, the time required to perform encryption has a negligible impact on the overall data processing time of the microcontroller, particularly when compared with the time needed for radio communication over the Wi-Fi network. Consequently, the overall efficiency of MQTT-based communication is preserved.

Furthermore, the results obtained using the modified MQTTset dataset show that enforcing fixed-length encrypted payloads does not significantly affect the effectiveness of most ML classification methods. Consequently, the ability to detect attacks and traffic anomalies in MQTT is also preserved. An important exception is the Naive Bayes classifier, whose performance significantly deteriorates after the introduction of encrypted payloads. This behavior results from the strong conditional independence assumptions underlying the Naive Bayes model which are violated by the statistical properties of encrypted data. Consequently, Naive Bayes classifiers appear unsuitable for intrusion detection in environments where MQTT payload encryption is applied.

From an application perspective, the results demonstrate that AES-based MQTT payload encryption can be integrated into smart home IoT systems without compromising their intrusion detection capabilities, provided that appropriate classifiers, such as tree-based or neural network models, are employed.

References

[1] EMQX Team, "MQTT with TLS: Fortifying MQTT Communication Security", 2023 (https://www.emqx.com/en/blog/fortifying-mqtt-communication-security-with-ssl-tls).

[2] N.T. Dhokane *et al.*, "S-MQTT: A Secure MQTT Protocol with Merkle Tree Authentication and AES Encryption for IoT Communication Systems", *Ingénierie des Systèmes d'Information*, vol. 30, pp. 1963–1973, 2025 (https://doi.org/10.18280/isi.300803).

[3] I. Vaccari *et al.*, "MQTTset, a New Dataset for Machine Learning Techniques on MQTT", *Sensors*, vol. 20, art. no. 6578, 2020 (https://doi.org/10.3390/s20226578).

[4] L. Ayavaca-Vallejo and D. Avila-Pesantez, "Smart Home IoT Cybersecurity Survey: A Systematic Mapping", *2023 Conference on Information Communications Technology and Society (ICTAS)*, Durban, South Africa, 2023 (https://doi.org/10.1109/ICTAS56421.2023.10082751).

[5] S. Lakshminarayana, A. Praseed, and P.S. Thilagam, "Securing the IoT Application Layer from an MQTT Protocol Perspective: Challenges and Research Prospects", *IEEE Communications Surveys and*

Tutorials, vol. 26, pp. 2510–2546, 2024 (https://doi.org/10.1109/COMST.2024.3372630).

[6] R.A. Mahajan *et al.*, "Enhancing MQTT Security in the Internet of Things with an Enhanced Symmetric Algorithm", *Journal of Electrical Systems*, vol. 20, pp. 126–137, 2024 (https://doi.org/10.52783/jes.758).

[7] A. Al-Ani *et al.*, "Evaluating Security of MQTT Protocol in Internet of Things", *2023 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, Regina, Canada, 2023 (https://doi.org/10.1109/CCECE58730.2023.10288857).

[8] H.Y. Chien, A.T. Shih, and Y.M. Huang, "Exploring MQTT Broker-based, End-to-end Models for Security and Efficiency", *Sensors*, vol. 25, art. no. 5308, 2025 (https://doi.org/10.3390/s25175308).

[9] I. Sahmi, A. Abdellaoui, T. Mazri, and N. Hmina, "MQTT-PRESENT: Approach to Secure Internet of Things Applications Using MQTT Protocol", *International Journal of Electrical and Computer Engineering*, vol. 11, pp. 4577–4586, 2021 (https://doi.org/10.11591/ijece.v11i5.pp4577-4586).

[10] M. Michaelides, C. Sengul, and P. Patras, "An Experimental Evaluation of MQTT Authentication and Authorization in IoT", *Proc. of the 15th ACM Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WiNTECH '21)*, pp. 69–76, 2021 (https://doi.org/10.1145/3477086.3480838).

[11] C. Sengul and A. Kirby, "RFC 9431. Message Queuing Telemetry Transport (MQTT). and Transport Layer Security (TLS). Profile of Authentication and Authorization for Constrained Environments (ACE) Framework", 2023 (https://doi.org/10.17487/RFC9431).

[12] J. Asharf *et al.*, "A Review of Intrusion Detection Systems Using Machine and Deep Learning in Internet of Things Challenges, Solutions and Future Directions", *Electronics*, vol. 9, art. no. 1177, 2020 (https://doi.org/10.3390/electronics9071177).

[13] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, „Survey of Intrusion Detection Systems: Techniques, Datasets and Challenges", *Cybersecurity*, vol. 2, art. no. 20, 2019 (https://doi.org/10.1186/s42400-019-0038-7).

[14] E. Jove *et al.*, „Intelligent One-class Classifiers for the Development of an Intrusion Detection System: The MQTT Case Study", *Electronics*, vol. 11, art. no. 422, 2022 (https://doi.org/10.3390/electronics11030422).

[15] M.A. Khan *et al.*, "A Deep Learning-based Intrusion Detection System for MQTT Enabled IoT", *Sensors*, vol. 21, art. no. 7016, 2021 (https://doi.org/10.3390/s21217016).

[16] A. Aqachtoul *et al.*, "MQTTEEB-D: A Real-world IoT Cybersecurity Dataset for AI-powered Threat Detection in MQTT Networks", *Data in Brief*, vol. 62, art. no. 111897, 2025 (https://doi.org/10.1016/j.dib.2025.111897).

[17] J. Aveleira-Mata *et al.*, "MQTT_UAD: MQTT under Attack Dataset. A Public Dataset for the Detection of Attacks in IoT Networks Using MQTT Protocol", *Data in Brief*, vol. 63, art. no. 112167, 2025 (https://doi.org/10.1016/j.dib.2025.112167).

[18] E.C.P. Neto *et al.*, "CICIoT2023: A Real-time Dataset and Benchmark for Large-scale Attacks in IoT Environment", *Sensors*, vol. 23, art. no. 5941, 2023 (https://doi.org/10.3390/s23135941).

[19] M. Sychra, "AESLib – Arduino and ESP AES library", *Arduino Docs*, 2023 (https://docs.arduino.cc/libraries/aeslib/).

Mariusz Gajewski, Ph.D.
Department of Cybersecurity
https://orcid.org/0000-0002-8084-6537
E-mail: M.Gajewski@il-pib.pl
National Institute of Telecommunications, Warsaw, Poland
https://www.gov.pl/web/instytut-lacznosci

Wojciech Sałabun, Prof.
Department of Advanced Information Technology
https://orcid.org/0000-0001-7076-2519
E-mail: W.Salabun@il-pib.pl
National Institute of Telecommunications, Warsaw, Poland
https://www.gov.pl/web/instytut-lacznosci
West Pomeranian University of Technology, Szczecin, Poland
https://www.zut.edu.pl